1986-12

An expert system to provide guidance on the operation of installed damage control systems aboard Naval ships in emergency situations

Gogel, Bernard G.

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

AN EXPERT SYSTEM TO PROVIDE GUIDANCE ON
THE OPERATION
OF INSTALLED DAMAGE CONTROL SYSTEMS
ABOARD NAVAL SHIPS
IN EMERGENCY SITUATIONS

by

Bernard G. Gogel

December 1986

Thesis Advisor Neil C. Rowe

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION Unclassified	1b RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY 2b DECLASSIFICATION / DOWNGRADING SCHEDULE	3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited
4 PERFORMING ORGANIZATION REPORT NUMBER(S)	5 MONITORING ORGANIZATION REPORT NUMBER(S)
6a NAME OF PERFORMING ORGANIZATION Naval Postgraduate School 6b OFFICE SYMBOL (If applicable) Code 52	7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School
6c ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000	7b ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000
8a NAME OF FUNDING/SPONSORING ORGANIZATION 8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c ADDRESS (City, State, and ZIP Code)	10 SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO / PROJECT NO / TASK NO / WORK UNIT ACCESSION NO

11 TITLE (Include Security Classification)
AN EXPERT SYSTEM TO PROVIDE GUIDANCE ON THE OPERATION OF INSTALLED DAMAGE CONTROL SYSTEMS ABOARD NAVAL SHIPS IN EMERGENCY SITUATIONS

12 PERSONAL AUTHOR(S)
Gogel, Bernard G.

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Master's Thesis	FROM ______ TO ______	1986 December	87

16 SUPPLEMENTARY NOTATION

17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Damage Control, Expert Systems

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

We discuss the design and implementation of Emergency, a prototype expert damage-control-guidance system. It provides recommendations on emergency action procedures, and information on the location and use of the installed damage control systems aboard the ship to Damage Control Central personnel, who can transmit the information to the On-scene leader at the site of the emergency. This prototype system handles emergencies involving personnel injuries, fire, flooding, or fumes. A type of decision lattice control structure was used for the program. The program takes advantage of the similarities that occur during all emergencies with general procedures at the top of the lattice. At the bottom of the lattice, procedures handle the detailed requirements for the identified hazard and source. This structure allows for future additions to or revisions of the methods of combating emergencies. The program Emergency has laid some valuable

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION Unclassified
22a NAME OF RESPONSIBLE INDIVIDUAL Neil C. Rowe	22b TELEPHONE (Include Area Code) 408-646-2462 22c OFFICE SYMBOL 52Rp

DD FORM 1473, 84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE
unclassified

#19 Abstract (continued)
groundwork for a prototype aboard a Naval ship.

Approved for public release; distribution is unlimited.

An Expert System to Provide Guidance on the Operation of Installed Damage Control Systems Aboard Naval Ships In Emergency Situations

by

Bernard G. Gogel
Lieutenant Commander, United States Naval Reserve
B.S., Purdue University, 1972

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
December 1986

ABSTRACT

We discuss the design and implementation of Emergency, a prototype expert damage-control-guidance system. It provides recommendations on emergency action procedures, and information on the location and use of the installed damage control systems aboard the ship to Damage Control Central personnel, who can transmit the information to the On-scene leader at the site of the emergency. This prototype system handles emergencies involving personnel injuries, fire, flooding, or fumes. A type of decision lattice control structure was used for the program. The program takes advantage of the similarities that occur during all emergencies with general procedures at the top of the lattice. At the bottom of the lattice, procedures handle the detailed requirements for the identified hazard and source. This structure allows for future additions to or revisions of the methods of combating emergencies. The program Emergency has laid some valuable groundwork for a prototype aboard a Naval ship.

THESIS DISCLAIMER

The reader is cautioned that the computer program developed in this research is only a prototype and may not have been exercised for all cases of interest. Many potential aspects of the the operation of installed damage control systems are not addressed by this program. While every effort has been made, within the time available, to ensure that the program is free of computional and logic errors, it cannot be considered validated. Any application of this program without additional verification is at the risk of the user.

TABLE OF CONTENTS

LIST OF FIGURES

I. INTRODUCTION

With the every increasing tendency toward the use of advanced and sophisticated equipment aboard ships in the US Navy it has become a problem maintaining expertise aboard the ships to operate them. Naval ships are provided with the most modern and effective types of fire fighting equipment it is practical and economical to use [Ref. 1: p. 175]. Installed damage control systems aboard modern, large ships can be very extensive and complicated. The operating environment of most U. S. Naval ships provides many opportunities for damage to occur to the ship and/or personnel. The installed systems aboard commercial ships and in large buildings have also gotten more complex and are understood by very few personnel. The ability to combat casualities in the most effective manner is essential to limit the damage to a ship or a building and to protect the personnel present. Proper use of the available installed damage control equipment is usually the most effective method to combat an emergency on a ship or in a building.

It is almost impossible for an individual to become an expert in all the various damage control systems. By the time the person is proficient in one or two major systems, he is ready for transfer or has reached the end of his obligated service. The locations of the controls and the valves of the installed damage control systems vary greatly on different ships. Personnel are not routinely assigned to the same class of ship when they are detailed for sea duty. Adequate training of shipboard personnel in the proper methods of handling emergency situations is an essential, but difficult requirement. Training should be done by the most competent damage control personnel available. Unfortunately the best people often are assigned other duties which greatly constrains their ability to take time to train others. Another element of training that is more difficult to teach is familiarity with the ship's layout and being able to find and operate the various systems on board [Ref. 2: p. vi]. Instead of attempting to teach personnel to become experts in the location and operation of the installed systems, it is proposed to develop an expert system to provide this information to the user.

An expert system uses some method of reasoning to eliminate bad choices and to determine the best course of action to acheive a goal. Expert systems use information

in an "intelligent" way to perform some task that is normally associated with human experts. There are many possible emergencies aboard a ship and there are many ways to combat them. Human experts in installed damage control systems aboard Naval ships are difficult to find or train. Some method to determine the best course of action for the given set of emergency conditions is needed. An expert system would be applicable for this use.

Numerous expert systems for many different applications have been developed and are in use today. DENDRAL is an expert system used by research chemists throughout the United States that identifies candidate molecular structures from mass spectral and nuclear magnetic data [Ref. 3: p. 51]. MACSYMA is a system used extensively by scientists and engineers to provide solutions for a variety of mathematical problems, such as algebraic simplification and integration [Ref. 3: p. 52]. MYCIN provides guidance in diagnosis and therapy in certain classes of infectious blood diseases [Ref. 3: p. 53]. CADUCEUS is a medical consultation system that attempts to make a diagnosis in the domain of internal medicine [Ref. 3: p. 40]. PROSPECTOR is an expert system that provides probabilistic interpretation of soil and geological data [Ref. 3: p. 54]. PROSPECTOR's performance is comparable to that of expert geologists. All of these systems enhance man's ability to make decisions after evaluating the many possible alternatives. The problem of providing assistance to personnel in the operation of installed damage control systems is similiar to some of the problems already solved with an expert system. It should be possible to develop an expert system to assist personnel in the identification and operation of installed equipment on a ship or in a building. This study will consider installed damage control systems on Naval ships.

This study discusses the design issues and the implementation of an expert system to provide guidance on the use of installed damage control equipment. This expert system, a prototype expert damage-control-guidance system, will be called Emergency in this study. The purpose of this thesis is to demonstrate that a computer program, designed as a type of expert system, to guide personnel in the operation of installed damage control systems is possible. In the event of a medical emergency, the program Emergency provides emergency first aid instructions and personnel casualty transportation precautions. In the event of a fire, flooding, or fumes, it provides a listing of berthing compartments to be evacuated and also a list of fittings and their location to be closed to set the primary boundaries. It will provide procedures to

determine the source of the hazard and provide safety precautions for the identified hazard. It lists the identification number and location of control devices to operate appropriate installed damage control equipment. It provides instructions on the proper and safe operation of the installed equipment to combat and eliminate the given hazard. The program also provides the proper procedures and tests to recover from the emergency.

Chapter II provides background on the typical damage control organizations aboard U. S. Naval ships. This includes how personnel are assigned to emergency teams, how personnel are trained, and the limitations of the current organization. Chapter III discusses the design and implementation of Emergency. There are some similarities between the methods used in handling the various kinds of emergencies, but each emergency also has the possibility of having some unique requirements depending on the location of the emergency in the ship. The control structure for the program must be able to make branching decisions to accomplish the detailed procedures for handling the emergency. Because of the need for branching from general to more detailed procedures, a design structure very similiar to a decision lattice was used. This type of design structure allowed the program to initially treat each emergency in the same general manner before implementing the detailed procedures associated with the identified hazard and source. Chapter IV provides a discussion of the methods of using, refining, and expanding Emergency for use aboard Naval ships. Chapter V contains the conclusions of the research effort involved in developing the prototype program. This section also discusses the limitations and and weaknesses of Emergency as well as the benefits. Appendix A is a user session listing demonstrating how Emergency handles medical emergencies. Appendix B is a user session listing demonstrating how Emergency handles a fire in a berthing compartment. Appendix C is a user session listing demonstrating how Emergency handles a main machinery room fire. Appendix D is a listing of program Emergency.

II. BACKGROUND

A. SHIPBOARD DAMAGE CONTROL ORGANIZATION

A Naval ship contains many potential hazards and can be a dangerous place under certain conditions. Emergencies such as fire, flooding, or fumes can occur at almost any time underway or inport. The most common shipboard emergency is fire. For this reason ships organize their primary emergency response team as a fire party. An emergency response team is a group of personnel from the ship that can perform a wide variety of tasks. The team would be made up of people with different talents that could be combined to combat emergencies. A ship must determine the availability of men and materials and design an organization for the employment of men and equipment to combat fires and other emergencies. Specific responsibilities, duties, and employment of equipment must be prepared and assigned to certain individuals, divisions, or departments. This information is put into a comprehensive and intelligible form called a "Fire Bill" and made available to all personnel addressed in the document. Standard ships' Fire Bills, established by each Type Commander and set forth in Type General Organization Books, are used as a guide by ships concerned [Ref. 4: p. 75]. The purpose of the Fire Bill is to establish a fire fighting organization and specify certain responsibilities for individuals and departments to ensure that fires and other related emergencies are effectively and quickly handled.

When the ship is underway, a special team is normally assigned to combat fires and other emergencies. The name of the unit, such as Fire Department, R-Division Flying Squad, etc., might vary from ship to ship, but the purpose is to provide a coordinated team of personnel well trained in firefighting and damage control that can respond quickly in an emergency. Most of the personnel assigned to this team are normally from the engineering ratings. Personnel in the engineering ratings work with the equipment that powers the ship. These personnel could be electricians, machinists, welders, pipefitters, boiler technicians, and others that work with their hands in a technical area. These personnel tend to have a good knowledge of general damage control procedures, but they are not particularity familiar with the location of all the installed damage control systems outside the engineering spaces where they work to maintain the propulsion plant and ship's service equipment such as electrical

generators. The engineering spaces are usually on the lower decks of a ship. Underway the Damage Control Assistant and senior damage control personnel will be available in Damage Control Central to provide what information is available from the damage control charts and manuals. The damage control charts and manuals in Damage Control Central contain all the information about all the ship's systems and all the compartments. A compartment, or space, on a ship is like a room or a corridor in a building. It has definable boundaries and is distinct from the surrounding spaces. The charts that show the compartments and installed systems are difficult to read because of the small print and the large volume of material on a single chart. Because of the small spacing, lines connecting the systems between compartments can be difficult to trace. Finding the watertight and/or flametight fittings to close in the event of a fire or flooding is also the responsibility of the personnel in Damage Control Central. The fitting are needed to set a complete firetight or watertight boundary at a single frame from one side of the ship to the other, both forward and aft of the emergency. These fittings can be doors, hatches, scuttles, or ventilation duct fittings. Some of the appropriate fittings can be easily missed when looking at a chart with hundreds of fittings. This information can take some time to determine from the manuals and charts. Once the required information is obtained, it is passed to the On-scene leader via the sound powered phone talker.

Fires or emergencies that occur during combat or while the ship is at general quarters should be handled as battle casualties by the Repair party organization in that section of the ship. The area covered usually extends from the damage control deck, normally the second deck, down to the keel, or bottom of the ship. Repair party personnel should be familiar with their area. In large ships, such as aircraft carriers, the area can become quite large. There is also a tendency for a large turnover of personnel assigned to repair parties. Therefore repair party personnel in many cases are not familiar with the location of all installed damage control equipment in their assigned area. This information is usually obtained from the damage control charts and manuals in the repair locker.

"The In-Port Fire Party shall be composed primarily of personnel in the regular damage control repair parties, each duty section having an effective fire fighting force" [Ref. 4: p. 75]. A typical Inport Fire Party aboard an aircraft carrier will consist of thirty men. If the ship has the normal six section inport watch rotation, this would result in 180 people in the Inport Fire Party organization at any given time. These 180

people, plus a number of alternates in the event of leave, schools, etc., have to be trained in their specific job and to work as a fire fighting team. "The time to train an individual in both general damage control and for a specific job on a fire team can take up to eight months depending on the individual's motivation and learning ability" [Ref. 5: p. 15]. It is almost impossible to train a significant number of personnel on the Inport Fire Parties on the location and operation of all the installed damage control systems.

B. SHIPBOARD DAMAGE CONTROL TRAINING ENVIRONMENT

The rapid increase in complex systems aboard Naval ships has made the damage control training task more difficult. Since crew members have to be proficient in the skills of their rating, damage control training will usually not have a high priority in the allotment of training time. The exercises that are held on board most ships only keep the members of the repair party organization familiar with general damage control procedures. By the time an individual crew member is familiar with the installed damage control systems on board the ship, he is ready for rotation or has reached the end of his obligated service. The general damage control knowledge that a crew member acquires and retains will be valuable for his future assignments. The knowledge acquired about installed systems is not as readily applied to future assignments since there is a great difference among ship classes as to the location of the control devices. The operating procedures for a given damage control system is the same or very similar for all ship classes. The procedures however can be very complex and it is difficult to memorize and retain a working knowledge of all of them.

C. METHODS TO AUGMENT THE KNOWLEDGE OF THE ON-SCENE LEADER

The key individual at the scene of an emergency is the on-scene leader. He must effectively coordinate the efforts of his fire party. LT. Stephen G. Weingart developed an Intelligent Computer Aided Instructional computer program to help train Fire Team leaders [Ref. 5: p. 21]. The use of Intelligent Computer Aided Instruction (ICAI) is a possible way to greatly improve the quality of training to handle damage control and emergency situations [Ref. 5: p. 21]. ICAI can be of great benefit in training personnel in general damage control and emergency procedures. The author, LCDR Gogel, prepared a study that focused on the development of an expert system that provides guidance to the on-scene leader in the proper utilization of the ship's

installed damage control equipment. The system would be extremely beneficial to the on-scene leader of the inport fire party, but would also be very valuable to the underway fire fighting organization. In the event of a battle casualty to the on-scene leader, the expert system could greatly assist an inexperienced on-scene leader take over the task of combating an emergency. This program is not a training system for the on-scene leader like LT. Weingart's program, but rather it is a tool to provide information that can be used by the on-scene leader. After several uses of the program the on-scene leader might learn all the detailed procedures for that situation. However, training is not the primary reason for this program. The primary purpose of the program is to provide an expert source of information on the operation of installed damage control systems.

Microcomputer systems that can store and retrieve large amounts of data in an accurate and rapid manner are becoming commonplace. "Computers have been proposed for use in the damage control organization to solve such problems as list removal, free surface effect, counterflooding, relocation of centers of buoyancy and gravity, and trim" [Ref. 2: p. 131]. A microcomputer with a database of information as to the proper number and the location of installed damage control systems would be an extremely valuable tool to augment the knowledge of the on-scene leader. The user would only have to enter information about the emergency in progress. The knowledge of the ship's systems and the proper procedures to operate them would be in the program or data base.

III. EMERGENCY: A DAMAGE CONTROL GUIDANCE SYSTEM

A. PROGRAM DESIGN CONSIDERATIONS

The problem of assisting the on-scene leader in the proper operation of installed damage control equipment was evaluated for the types of emergencies that could be anticipated. The author of this study has been involved in all aspects of damage control aboard aircraft carriers. That experience and the guidance of the appropriate Naval Sea Systems Technical Manuals were used in evaluating the needs of the on-scene leader to combat an emergency. It was determined that at the beginning most emergencies require similar courses of action. Emergencies can be divided into the major areas of injury to personnel or damage to the ship. Emergencies are reported at a given location using the ship's compartment numbers. Figure 3.1 shows how the report of an emergency should be initially evaluated. If the emergency was a personnel injury, it should be handled as shown in Figure 3.2 . If the emergency was a fire, flooding, or fumes, many similar actions should take place. Boundaries should be set to contain the fire, flooding, or fumes to one area of the ship. The source of the emergency should be determined. The source or cause of the fire, flooding, or fumes should be eliminated. Procedures to recover from the emergency should be initiated. Figure 3.3 shows the sequence of actions that need to be taken in the case of a fire, flooding, or fumes. The details of the actions would vary depending on the type of emergency.

Even though many of the basic procedures for handling emergencies are the same, there are a large number of possible combinations of hazard types and sources. The methods of handling emergencies will vary depending on the location in the ship and what installed systems are available. A library of procedures for different cases would not be practical due to the large number of possible cases that would have to be covered. An expert system that, for specific hazards, can determine the applicable segments of general procedures was needed.

A compiled program was desired for efficient operation of the program on a microcomputer on various ships at sea. "Compilers take a program in an easy-to-read but slow-to-execute form and convert it to a more efficient one" [Ref. 6: p. 95]. "Compilation techniques can also make many kinds of artificial intelligence programs more efficient" [Ref. 6: p. 95]. It was essential that the program be as general as

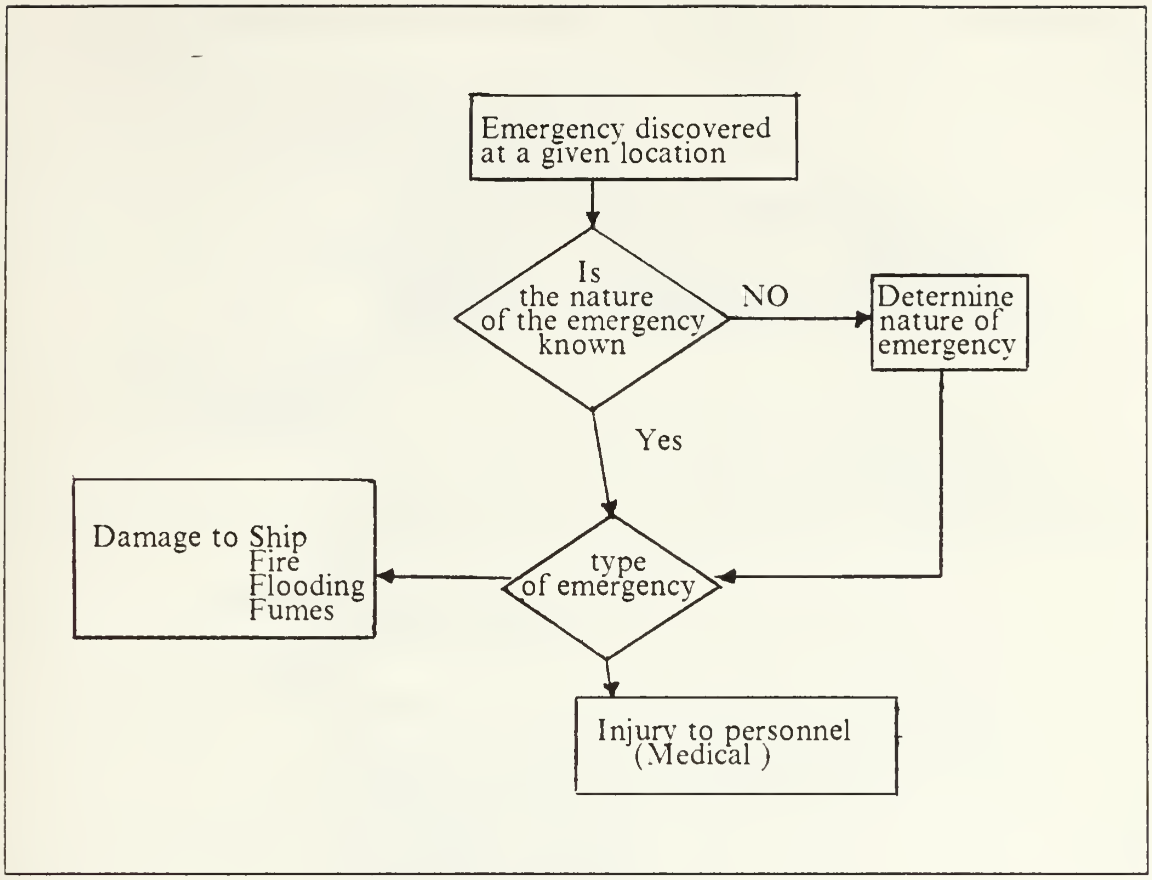

Figure 3.1 Initial evaluation of a report of an emergency.

possible and use common procedures where applicable. Good programming design was desired also for ease of maintaining and improving the program. The use of a decision lattice was considered as a possibility. A decision lattice would permit a restricted but efficient form of forward chaining.

The prototype program was designed using a top-down stepwise refinement methodology. The overall design was a type of decision lattice structure with some global variables to pass the type and the source of the hazard. The program was designed to obtain the data about the installed systems aboard the ship from disk files. Information on the type and location of the emergency is to be entered at the terminal keyboard. Case structures, as shown in Figure 3.4, are used to make decisions as to the next recommended course of action to present to the user. When the program cannot resolve a question by inference from what it already knows, it asks the user to provide information that it needs. This information is used with information about the

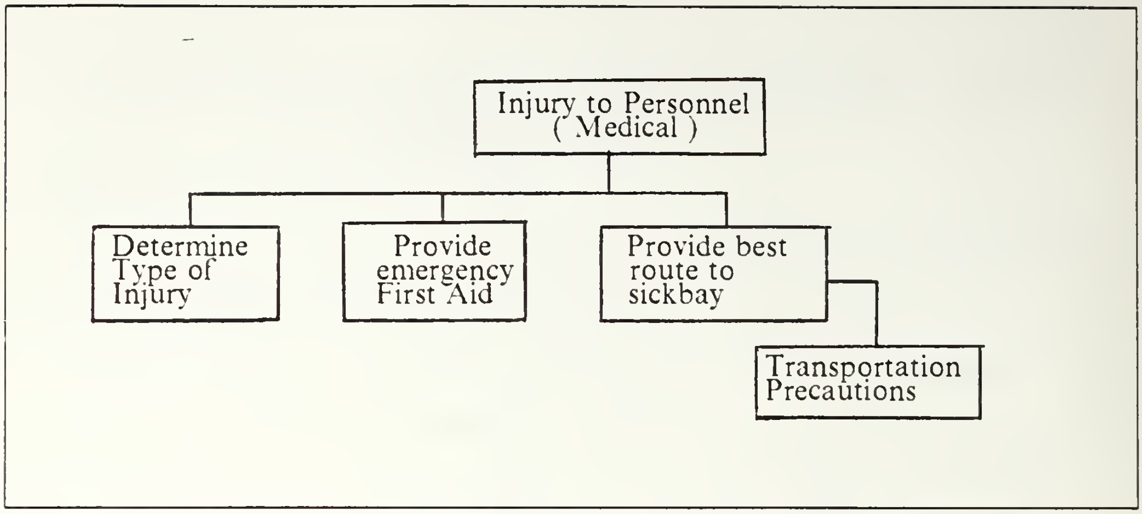

Figure 3.2 Procedure to handle a personnel injury.

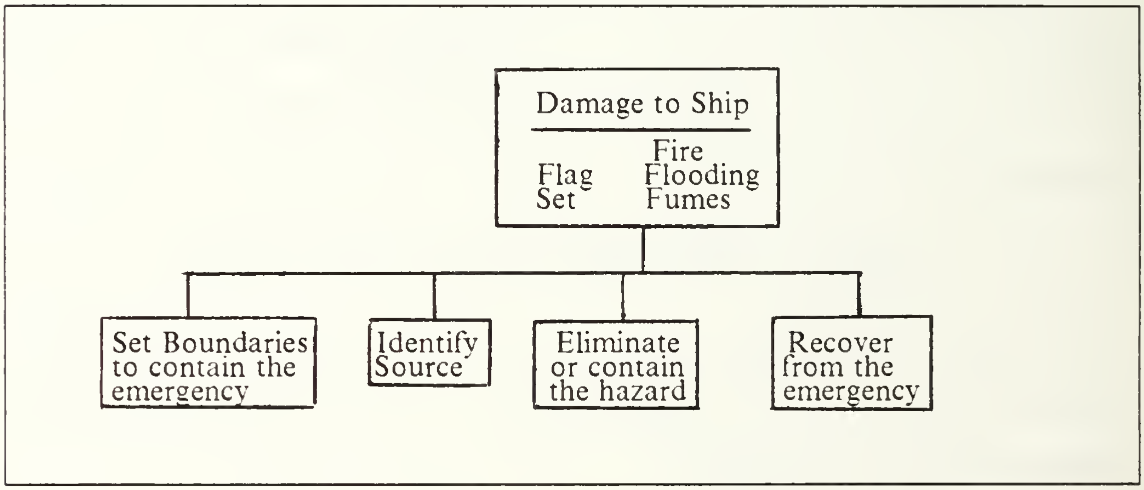

Figure 3.3 Sequence of actions in the event of damage to the ship.

compartment where the emergency is located to determine the best recommendations to pass to the user.

B. EMERGENCY SYSTEM: DESCRIPTION OF OPERATION

Emergency is a program in which the typical user is a sailor on duty, or in Naval terms standing a watch, in Damage Control Central who must pass information to the On-Scene Leader at the scene of an emergency. Even on larger ships, some of these

```
case needrec@.instalsystems of
  6 : begin
         if hazsource = 1 then
             writeln('Use fire station to combat class "A" fire',
                                              ' if possible .');
         if hazsource = 4 then
              writeln('Use CO2 if appropriate to ',
                            'extinguish the class "D" fire.');
         getco2hr(needrec);
      end;  (* 6 *)

  7 : begin
         writeln(' Use the HALON system if the fire is out',
                                    ' of control or great');
         writeln(' hazards to personnel are involved in',
                               ' trying to combat the fire .');
         gethalon(needrec);
      end; (* 7*)

 10 : if hazsource < 3 then
          getafffhr(needrec);

 11 : begin
         writeln('The compartment contains various ',
                             'firefighting systems.');
         if hazsource = 4 then
             writeln(' Use TAFES to extinguish class "D" fire',
                          ' if appropriate.');
         if hazsource <> 3 then
             gettafes(needrec);
         if hazsource < 3 then
              getafffhr(needrec);
         gethalon(needrec);
      end;  (* 11 *)
end;  (* case instalsystem of *)
```

Figure 3.4 Example of a case structure.

watchstanders lack the experience to obtain the proper information in a timely manner to pass on to the On-Scene Leader. The program Emergency provides that assistance and guidance.

Emergency is designed to start with the report of an emergency at a given location on the ship. This is how an emergency response would start on board an actual ship. The author used his years of experience in damage control aboard ships, including aircraft carriers, to design the order of actions that need to occur to combat a given emergency. The procedures to follow in combating the emergencies were obtained from experience and from guidance from Naval Sea Systems Command

Technical Manual, Chapter 079 [Ref. 1], and Naval Sea Systems Command Technical Manual, Chapter 9930, Firefighting Ship [Ref. 4]. For this prototype implementation, the user is provided a listing of the ship's compartments. This option gives the user a list of valid compartment numbers as shown in Figure 3.5 to test the program. After

The Following Compartments will work as input for the program:

Compt. number	Compt. Use	Installed Systems
01-50-3-L	Berthing Compartment	None
02-50-3-L	Passageway	None
04-40-0-L	Control Space	None
2-30-2-Q	Workcenter	Dry Standpipe Spk
2-30-4-Q	Passageway	None
2-30-1-Q	Workcenter	CO2 Hose Reel
2-30-5-Q	Admin Space	None
2-30-6-L	Head Space	None
2-36-2-Q	Repair Locker	None
2-37-1-Q	Passageway	None
2-40-0-L	Galley Space	None
2-48-2-Q	Galley Space	Dry Standpipe Spk
2-50-1-Q	Workcenter	None
2-60-0-L	Wardroom/galley	None
2-60-2-Q	Passageway	None
2-60-3-Q	Admin Space	None
2-60-6-Q	Medical Space	None
3-30-3-L	Head Space	None
3-60-7-L	Head Space	Exhaust Fan
3-70-2-L	Berthing Compartment	None
3-70-4-L	Head Space	Exhaust Fan
3-73-1-Q	Passageway	None
3-80-0-L	Berthing Compartment	None
3-90-3-E	Cominuter Space	Installed Eductor
4-30-1-A	Storeroom	Dry Standpipe Spk
4-30-2-M	AMMO Magazine	Active Sprinkler Sys
4-80-0-C	Control Space	None
4-80-6-Q	Passageway	None
4-80-4-A	Storeroom	Exhaust Fan
5-30-4-M	AMMO Magazine	Active Sprinkler Sys
5-30-1-A	Storeroom	Dry Standpipe Spk
5-40-0-E	Main Machinery Room	HALON, TAU, CO2 HR
5-60-0-E	Main Machinery Room	Halon, TAU, CO2 HR
6-30-1-A	Flammable Storeroom	CO2 Flooding
6-30-2-E	Pump Room #1	AFFF HR and CO2 HR
6-30-3-A	Storeroom	Dry Standpipe Spk
6-80-0-E	Pump Room #2	AFFF HR and CO2 HR

Figure 3.5 List of possible Compartments for Emergency.

the compartment number is entered, the user is queried if the nature of the emergency is known. If unknown, the program provides some guidance as to how to determine the nature of the emergency. Once the nature of the emergency is known the user selects

one of the four types of emergencies from a menu shown in Figure 3.6 . The emergency will be either a personnel injury or some damage or danger to the ship and personnel. A global variable, *haztype*, is set to either fire, flooding, fumes, or personinj depending on the number selected. This global variable is used to determine how the program will execute the details of the common procedures to accomplish the required actions for the specified hazard.

```
What is the nature of the emergency or problem ?
Enter the appropriate number below :

      Fire                              = 1
      Flooding                          = 2
      Fumes                             = 3
      Injury to personnel               = 4
```

Figure 3.6 Menu for the user to select the type of emergency.

If the emergency is a personnel injury, the user is asked to enter the type of injury from a list of possible injuries shown in Figure 3.7 Using the inputted type of injury, the program will provide the user with recommendations for emergency first aid

```
What type of medical emergency has occurred ?
Enter the appropriate number below :

      Head Injury                       =  1
      Back or neck Injury               =  2
      External Bleeding                 =  3
      Internal Injury                   =  4
      Broken Leg                        =  5
      Broken Arm                        =  6
      Severed Limb                      =  7
      Serious Burns                     =  8
      Inhalation of Fumes               =  9
      Electrical Shock                  = 10
      Drowning                          = 11
      Unknown Illness                   = 12
```

Figure 3.7 Menu to select type of Injury or Illness.

and precautions for the safe transportation of the casualty to the ship's sick bay.

If the emergency is fire, flooding, or fumes the user is presented with a list of berthing compartments, if applicable, to be cleared. The compartments are determined by checking the use code in the compartment records of all compartments in the same zone or adjacent zone to the compartment that has the emergency. If the use code is for a berthing space the compartment is listed. The user is then given the location and number of the electrical panel and the ventilation controller to secure the power and ventilation to the space. The user is then presented with a list of water-tight and fume-

Set the following Boundaries to contain the Hazard:

Fitting No.	WTD 4-37-1	located at	4-37-1-Q
Fitting No.	WTH 4-37-3	located at	4-37-1-Q
Fitting No.	WTS 4-39-1	located at	4-37-1-Q

Figure 3.8 Notice of Boundaries to be set.

tight fittings, as shown in Figure 3.8, to close to isolate the hazard to one area of the ship. If the source of the emergency is not known, the user is given suggestions as to some possible sources in the area, compartments to check, piping systems to consider, and tests to conduct. The user is presented with safety precautions for the identified

Hydrogen Sulfide is highly toxic. All personnel who might be exposed in the course of finding and eliminating the source of the fumes should use breathing devices.

Figure 3.9 Example of a Safety Precaution.

hazard. Figure 3.9 shows an example of a typical safety precaution. If no appropriate installed damage control systems exist, the user is given recommendations on what portable equipment to use. If installed systems are appropriate for the emergency, the user is presented with a list of the control devices and their location. An example for a HALON system is shown as Figure 3.10 . Operating instructions and safety

The compartment's HALON Actuators are as follows:

Actuator act# 4-40-2 located 4-40-0-C FR 42 P
Actuator rmact# 2-40-1 located 2-40-1-Q FR 40 P

Figure 3.10 Fitting locations for a HALON System.

precautions for the installed equipment are then presented. An example for an installed HALON system is shown in Figure 3.11 . In the event of fumes or when a fire is extinguished, recommended atmosphere tests are presented. In the event of a fire, when

Before operating the installed HALON system , all supply and exhaust vent dampers should be closed if possible, and all fans serving the space secured. All personnel should be out of the space or in the process of leaving the space. To activate the HALON system use the given local or remote pneumatic actuators

Figure 3.11 Operating Instructions for a HALON System.

the fire is overhauled additional tests and recovery procedures are then presented to the user. If needed and appropriate, control devices and their location are given for the operation of installed ventilation and dewatering equipment. If there are no installed ventilation or dewatering systems in the compartment, the program will provide suggestions on what portable equipment to use.

C. DATA STRUCTURES

The problem of how to access the large amount of possible data concerning the compartments on the ship was studied. For efficient operation the program should not be spending a lot of time obtaining data on the compartment or emergency in question. The required information needs to reside in memory in a data structure that allows quick access to any required record. The method by which compartments are numbered on board ship does not lend itself to the sequential use of compartment

numbers to find particular data. The use of one large record per compartment that contained all possible data about that compartment was considered. A large ship has many compartments and very few, if any, contain all of the various damage control equipment. In fact most compartments do not contain any installed damage control equipment. Some of the data required for a compartment such as electrical panel controlling the power and the ventilation controller controlling the supply ventilation is the same for a number of compartments in the area. For each record to contain a field for a control number and location for each type of installed equipment would be wasteful and inefficient. The use of record variants was considered [Ref. 7: p. 330]. Since most of the ship's compartments do not have any installed damage control equipment, it was decided that this would still be inefficient. Some compartments do contain more than one type of installed damage control equipment. It was decided that several different data structures were needed. A record for each compartment on the ship was required to store and retrieve certain critical information to isolate and contain the emergency. The location of electrical power panels to secure electrical power, vent controllers to secure supply ventilation, and the fitting numbers and their location as appropriate are needed as soon as the location of the emergency is identified. Some of the data required, such as fittings to close to set boundaries, is quite extensive and is the same for a number of compartments. Boundaries are located at frames in the ship where a watertight or flametight bulkhead extends across the entire width of the ship from port to starboard. To design the program, all compartments, on the same deck or level, that are between two sets of boundaries are considered to be in the same zone. The boundaries to prevent the spread of the hazard would be the same for all compartments in the zone. It was decided to use a code number for all systems that cover more than one compartment. Figure 3.12 shows the codes for the piping systems that could be present in a compartment. There are addition possibilities for piping systems aboard a ship, but for the purposes of this prototype program only these combinations were considered. Codes were also developed for the types of compartment on the ship as shown in Figure 3.13 . Codes for the considered installed damage control systems are shown in Figure 3.14 .

In order to find such data as what berthing areas to clear, compartments to check to determine source of hazard, etc., an ability to find information about compartments in the vicinity of the emergency was necessary. To be efficient a method to limit the search of the records of the compartments was needed. By organizing the

Piping Code #	Type of Piping System or systems
0	NONE
1	Fire main
2	Freshwater
3	Fire main and Freshwater
4	Chilled Water
5	Soil Drain
6	Plumbing Vent
7	High Pressure Air
8	Fuel Oil
9	Fire main and Fuel Oil
10	Fuel Oil and Soil Line
11	Hydraulic Oil
12	Multiple Systems

Figure 3.12 Codes for Piping Systems.

Use Code #	Use of Compartment
1 :	Admin space
2 :	Berthing Compartment
3 :	Passageway
4 :	Head space
5 :	Workcenter
6 :	Main Machinery Room
7 :	Boiler Room
8 :	Pump room
9 :	Emergency Generator
10 :	Comminuter space
11 :	Galley space
12 :	Storeroom
13 :	Flammable storeroom
14 :	Fuel tank
15 :	Water tank
16 :	Void
17 :	Access Trunk
18 :	Control space
19 :	Fan Room
20 :	Repair Locker
21 :	Hazardous materials
22 :	AMMO Magazine
23 :	Medical Spaces

Figure 3.13 Codes for Compartment Use.

Code #	Installed System or Systems
0 :	NONE
1 :	Installed eductor
2 :	Installed exhaust fan
3 :	Active sprinkler system
4 :	Dry standpipe sprinkler
5 :	CO2 Flooding system
6 :	CO2 Hose Reel
7 :	HALON
8 :	Twin Agent Unit or TAFES
9 :	AFFF Sprinkler
10 :	AFFF Hose Reel
11 :	HALON, TAU, and CO2 Hose Reel
12 :	AFFF HR and CO2 Hose Reel

Figure 3.14 Codes for Installed Damage Control Systems.

records of compartment data in a linked list of records, compartments in the same zone were kept together. All the records for the compartments on a deck or a level are in a single linked list. The linked lists are in an array by deck or level. The desire to make the user's entry format for the ship's compartment number as close to the actual compartment number as possible, had a great impact on the design of the data structure for the compartment records. Decks on a ship are numbered 1, 2, 3, etc. starting at the main or highest complete deck on the ship and going down thru the lowest deck at the keel. For this prototype program the 6th deck is at the keel. Levels above the main deck are labeled 01, 02, etc. up to the highest level on the ship. The program needs to handle the deck or level number as an integer in order to find the required information on a given compartment and to determine what compartments, if any in the area might need to be evacuated. Since an 01 and a 1 would both be interpreted as the integer 1, a boolean field in the compartment record is set to indicate if the compartment is above the main deck. A separate array of linked lists of records is used for the compartments above the main deck and another array of linked lists of records is used for the compartments on the main deck and below. This allows the program to access the right compartment.

The records for the compartments contain fields for the detailed location of the compartment, codes to secure the power and ventilation, code for what zone it is in, code for the use of the space, a code for the piping systems that pass through the

space, and a code for the installed systems, if any, in the space. By checking the appropriate fields in the record of the compartment where the emergency is located, the program can determine what systems are present and what the space is used for. In the event the compartment has more than one installed damage control system, the program recommends the most appropriate system for the given emergency. The program has the ability to identify all other compartments in the same zone as the emergency. It can determine if there are certain types of compartments that could be dangerous with the given emergency in the area.

IV. USING THE EMERGENCY SYSTEM ON US NAVY SHIPS

A. MEMORY REQUIREMENTS

To use the Emergency system on a microcomputer aboard US Naval Ships, the memory requirements for the system's files must not be excessive. Emergency was developed on a COMPAQ Deskpro computer with 640K of memory and a 10 MB internal hard disk drive. The program Emergency and the prototype data files all fit on a standard 5.25 inch, one-sided, double density floppy disk. Emergency uncompiled takes 82K bytes of disk space and compiled takes 47K bytes. For the prototype program the data files consume approximately 8K bytes of memory. The prototype data files only include 48 compartments and a large aircraft carrier might contain approximately 1200 compartments. It is estimated that the data files for an aircraft carrier would consume 200- 300K bytes of memory. The compiled program Emergency and the data files for most Naval ships would fit on one standard 360K floppy disk. Due to its large size and many complex systems, a separate disk for the data files might be required for an aircraft carrier. The exact volume of data required for any given ship type is not known.

B. PROGRAM LANGUAGE REQUIREMENT

The Emergency system was developed on a COMPAQ Deskpro computer in TURBO PASCAL as described in Borland International Turbo Pascal Version 3.0 Reference manual [Ref. 8]. The system files and the prototype data files were then transferred to a VAX 11/780 UNIX file system and Emergency was modified to run on UNIX with Berkeley Pascal as described in the Berkeley Pascal User's Manual [Ref. 9]. The Turbo Pascal version worked better than the Berkeley Pascal version because Turbo Pascal was designed for use with microcomputers. Turbo Pascal has a built in ability to read string variables and no special procedures are needed to handle strings. With Berkeley Pascal additional procedures or functions have to be created to read in string variables.

ADA will be the Department of Defense computer language [Ref. 10: p. 269]. ADA is a Pascal like language and anything written in Pascal could be converted to ADA. Emergency could be converted to ADA to run aboard US Naval ships.

C. HOW TO INSTALL EMERGENCY

Emergency is a prototype program and was not designed for one particular ship. Emergency was designed to be easily installed in all Naval ships. The only part of the program that needs to be tailored for the individual ship is the value of some of the constants that give the number of decks, electrical panels, etc. and the constants, such as the compartment number of sickbay, that provide specific information where a few special compartments are located on the ship. Compartment numbers for the regular compartments will be in the data base. The program constants are conveniently located at the beginning of the program. The information for the constants can be obtained from the ship's damage control drawings or from the damage control manuals.

The unique part of the Emergency system for each ship is the data base. Preparing and installing the data base is the difficult and time consuming part of installation. It is extremely important that the data base be accurate because this is where the program obtains the information about the ship's systems. The major file is the 'Compdata' file which contains all the coded information for each compartment on the ship. There are also thirteen (13) data files containing information on the identification numbers and locations of such items as electrical panels, ventilation controllers, water-tight and flame-tight fittings, and installed damage control systems. For the actual implementation on board Naval ships, certain data fields in certain files might have to be expanded to contain the required information. These files can be in a standard format for all ships, but the data in each file would be unique for the ship. Ships in the same class could have similar data that could be shared with each other. Ships in a class could have data passed down from the lead ship of the class. This would greatly simplify the preparation of the data bases. Many of the installed damage control systems on the newer classes of ships are virtually identical between ships in the same class. The layout and use of the compartments are also very close. These elements of data could be inherited from the class design concept by all the ships in the class. It might be possible to have Type Commanders, who are administratively in charge of all the ship's of a given type, prepare the data base for these classes of ships under their control. In the event the ship's force has to enter the data base, the most important compartments and systems should be entered first. Emergency can then function in a partial mode but it would not cover the complete ship or give complete information. For new construction contracts, a provision could be made that the ship builder prepare the data base for the ship under construction since they would have all

the plans and specifications. The ship would then be covered by the system upon completion of construction.

Emergency is designed to be used on a microcomputer aboard Naval ships. The primary location for the installation of the Emergency system should be Damage Control Central. This is the location where the Damage Control Assistant underway or the duty Engineering Officer in-port will work with the Damage Control watchstander to pass information via a sound powered phone talker to the On-Scene leader at the scene of the emergency. As a backup, another microcomputer to use with Emergency could be installed in secondary Damage Control Central. General Quarters is when the ship goes to its maximum state of readiness to combat the enemy or combat inflicted battle damage. Groups of trained personnel are organized as repair parties to combat any damage or emergencies. The repair party meets at a compartment that contains portable damage control equipment. This compartment is called a repair locker. On large ships a microcomputer with the Emergency system would be valuable at all the major repair lockers to use during General Quarters. This would provide a means to assist new on-scene leaders to combat damage to the ship in the event of battle casualties to personnel in the repair party.

Training of damage control central personnel to operate Emergency would be easier than trying to train them to read and interpret the complicated and numerous damage control drawings. The program would present them only with the information that they need and eliminate the possibility of the wrong system being traced out from a damage control drawing. With an installed hard disc, the microcomputer could be set up with a batch file to call Emergency whenever the computer is turned on. The operator would then have to enter the compartment number of the location of the emergency in the format shown on the screen. All other entries would be of a yes/no nature or a selection from a menu. Training of personnel to update the data base or maintain the program would take more time. Since the data base only changes when alterations to the ship's design are made, very few people would be needed. With the rapid increase in computer literacy, every ship would probably have several people who could perform this function.

To test the emergency system, there is an option at the beginning of the program to list all of the ship's compartments. An operator can select a compartment, determine its use, and determine what installed damage control systems it contains. The operator can then enter the compartment number and see if the expected boundaries,

system controls and locations, and procedures for the selected hazard type are presented.

V. CONCLUSIONS

A. DISCUSSION OF THE RESEARCH

This study tried to address the problem of lack of expertise in the operation of installed damage control systems aboard Naval ships. Evaluation of the problem resulted in the decision to use an expert system. The work involved in developing the prototype expert system Emergency was a success in that Emergency was capable of performing some nontrivial examples of providing information on emergency procedures and the operation of installed damage control systems. Appendixs A, B, and C of this study are sample user session listings that demonstrate the operation of the program. The use of a decision lattice structure with some global variables to pass the type and source of the hazard proved to be an efficient method of designing the program. Common upper-level procedures could be used for all the different kinds of emergencies, but the unique requirements for the different emergencies could be handled within the procedures by testing the value of the global variables and determining the type of compartment where the emergency is located. The use of the Pascal language to write an expert system for the operation of installed damage control equipment was shown to be feasible by the prototype program. Chapter IV showed that implementing Emergency aboard Naval ships was possible. It is proposed that expert systems for use aboard naval ships can be designed with a type of decision lattice structure using the Pascal or ADA language.

B. EMERGENCY SYSTEM LIMITATIONS

Emergency is a prototype program that was designed to demonstrate the concept that an expert system could provide valuable assistance in the employment of installed damage control equipment. The prototype program only handles emergencies in one compartment at a time. The program would be of greater value, if several emergencies located in different compartments could be handled at the same time. This is a possible future upgrade to the program.

Due to time limitations most of the procedures that provide guidance to the user are not developed fully. As an example, the procedure *perinjury* contains procedures that handle personnel causalities. On the menu displayed to the user, see figure 3.7,

only twelve possible injuries are listed. Of the twelve choices given to the user, only two procedures for emergency first aid are partially developed. The other medical procedures are merely stubs and need to be developed further. The procedures *Tranport* and *bestroute* which provide transportation precautions and the best route to sickbay would be more useful if they were developed further. An expert in Emergency first aid should be consulted in expanding and improving the procedures involving personnel causalities.

All of the current, major installed damage control systems were addressed to some extent in Emergency. Some of the Technical Data on system operations is very limited due to the unavailability of some of the Technical Manuals. The safety-precaution procedure and the procedure to provide system guidance need to be improved to include more information from the Naval Sea Systems Command Technical Manuals. The fume-test procedure needs to include tests for more types of fumes and additional information on when to conduct what tests. The procedure for an unknown hazard needs to be developed to provide the user with a list of compartments to check and tests to conduct to determine the nature of the emergency. These procedures will have to be improved and expanded prior to actual use of Emergency in the fleet. This is a time consuming task which the author of this study did not have time to pursue. The author estimates that the most important procedures are 75 to 80 percent complete on the information they present to the user. The procedures that present safety precautions and system operating guidance are estimated to be only 40 to 50 percent complete. All procedures are complete enough to show how they interact with the user and the other procedures. The author estimates that it would take three manmonths to complete the program as a useful prototype for a Naval ship.

The Emergency program was developed using a limited number of compartments rather than data for an entire ship. With this number of compartments it was possible to demonstrate the feasibility of the concepts without handling the large amount of data a real ship would have entailed. The drawback of the small data base is that the ability of the program to handle large amounts of data read in from a disk file has not been fully demonstrated. The input from the user at the keyboard will remain very limited regardless of the number of compartments the ship contains. There is nothing in the program and data structure design that should preclude the proper handling of large amounts of coded data from disk files.

The decision lattice control structure was not limiting in any way. The structure allowed a lot of flexibility in arranging the procedures. The program was written in Pascal which did place some constraints on how the procedures were arranged, but this was not a problem since the procedures worked together better in a structured format. I would recommend the use of Pascal for other expert systems that use a decision lattice structure.

C. FUTURE PROGRAM IMPROVEMENTS

Emergency was written to be easily used by nontechnical personnel. The entry of the compartment number is the only input the user must make that requires more than the entry of a character (Y/N) or an integer choice from a menu. In spite of this there could be some advantages in installing a graphics interface for the program. A graphics interface that displayed a drawing of the ship's compartments by deck or level would give the user a prospective of the relationship of one compartment to another. By the use of a mouse the user could select the compartment of interest where the emergency is located. Pop up or drop down menus could be used to select the type of emergency. A helpful addition to the program would be a graphic display that showed the relative location of all the installed damage control systems control operators and their identification numbers. The graphic display would have to be well designed to avoid confusing inexperienced users with too much data as the present damage control charts tend to do. The installation of a graphical interface would greatly increased the installation time to developed the data base and would require increased data storage area.

D. BENEFITS OF RESEARCH

This study demonstrated the feasibility and benefits of using an expert system to assist the damage control organization aboard naval ships in the proper use of installed damage control systems. The advantage of the decision lattice control structure in conjunction with the use of some global variables was demonstrated. The use of Pascal as a practical language for expert systems was shown. The use of Pascal demonstrates that ADA is also a possible language for expert systems.

This research can also benefit other areas where an expert system might be beneficial.

1. An expert system might be beneficial in determining the source of a boiler water chemistry problem.

2. Engineering plant casualty control could be assisted with an expert system. In a sense a manual version of an expert system already exists with the check lists for given casualties in the Engineering casualty control manuals. This system could be automated.

3. The developed prototype expert system could be converted into an Intelligent Computer Aided Instruction (ICAI) system to train personnel in the operation of the installed damage control systems.

APPENDIX A

DEMONSTRATION OF EMERGENCY PROGRAM HANDLING A MEDICAL EMERGENCY

==

The following is a demonstration of the program Emergency handling a medical emergency. The compartments covered by Emergency were printed out first. The program's interaction with the user is shown.

==

```
***********************************************************
*                  EMERGENCY                         *
* An Expert system to provide guidance on the operation of *
* installed damage control equipment aboard the ship.      *
* Answer the given questions and make the appropriate      *
* choices from the presented menus.                        *
***********************************************************


Do you want to have the ship's compartments printed out ? Y/N

y
The following compartments will work as input for the program.


Compt number        Use              Installed systems

 01- 50- 3-L   Berthing Compartment  : NONE
 02- 50- 1-Q   Passageway            : NONE
 04- 40- 0-C   Control space         : NONE

  2- 30- 2-Q   Workcenter            : NONE
  2- 30- 4-Q   Passageway            : NONE
  2- 30- 2-Q   Workcenter            : Dry standpipe sprinkler
  2- 30- 1-Q   Workcenter            : CO2 Hose Reel
  2- 30- 6-L   Head space            : NONE
  2- 30- 5-Q   Admin space           : NONE
  2- 30- 3-Q   Passageway            : NONE
  2- 36- 2-Q   Repair Locker         : NONE
  2- 37- 1-Q   Passageway            : NONE
  2- 40- 1-Q   Admin space           : NONE
  2- 40- 2-Q   Galley space          : NONE
```

2- 40- 0-L	Galley space	: NONE
2- 48- 2-Q	Galley space	: Dry standpipe sprinkler
2- 50- 1-Q	Workcenter	: NONE
2- 60- 6-Q	Medical Spaces	: NONE
2- 60- 3-Q	Admin space	: NONE
2- 60- 2-Q	Passageway	: NONE
2- 60- 1-Q	Passageway	: NONE
2- 60- 0-L	Galley space	: NONE
2- 68- 2-L	Head space	: NONE
2- 70- 1-Q	Admin space	: NONE
2- 76- 1_Q	Passageway	: NONE
3- 30- 3-L	Head space	: NONE
3- 30- 2-L	Berthing Compartment	: NONE
3- 60- 7-L	Head space	: Installed exhaust fan
3- 70- 4-L	Head space	: Installed exhaust fan
3- 70- 2-L	Berthing Compartment	: NONE
3- 73- 1-Q	Passageway	: NONE
3- 80- 0-L	Berthing Compartment	: NONE
3- 90- 3-E	Cominuter space	: Installed eductor
3- 90- 3-Q	Workcenter	: Installed exhaust fan
4- 30- 1-A	Storeroom	: Dry standpipe sprinkler
4- 30- 2-M	AMMO Magazine	: Active sprinkler sys
4- 80- 1-Q	Passageway	: NONE
4- 80- 0-C	Control space	: NONE
4- 80- 6-Q	Passageway	: NONE
4- 80- 4-A	Storeroom	: Installed exhaust fan
5- 30- 1-A	Storeroom	: Dry standpipe sprinkler
5- 30- 4-M	AMMO Magazine	: Active sprinkler sys
5- 40- 0-E	Main Machinery Room	: HALON, TAU, and CO2 hosereel
5- 60- 0-E	Main Machinery Room	: HALON, TAU, and CO2 hosereel
6- 30- 1-A	Flammable storeroom	: CO2 Flooding
6- 30- 2-E	Pump room	: AFFF HR and CO2 hosereel
6- 30- 3-A	Storeroom	: Dry standpipe sprinkler
6- 80- 0-E	Pump room	: AFFF HR and CO2 hosereel
6- 90- 0-A	Hazardous materials	: CO2 Flooding

Give the compartment number of the location of the emergency .
Enter the compartment number in format __ ___ _-_
Example : 5 60 0-E ; NOTE: The spaces are important.
If the exact location is unknown give the closest location known.
The format must be accurate. For example "1-Q " vice "1-q" .
4 30 2-M

The compartment submitted is 4- 30- 2-M

Is the nature of the emergency known ? Y/N

y

What is the nature of the emergency or problem ?
Enter the appropriate number below :

Fire = 1
Flooding = 2
Fumes = 3
Injury to personnel = 4

4

What type of medical emergency has occurred ?
Enter the appropriate number below :

Head Injury = 1
Back or neck Injury = 2
External Bleeding = 3
Internal Injury = 4
Broken Leg = 5
Broken Arm = 6
Severed Limb = 7
Serious Burns = 8
Inhalation of Fumes = 9
Electrical Shock = 10
Drowning = 11
Unknown Illness = 12

8

Try to limit contact and keep the burned area as clean as possible. Have the Hospital corpman apply temporary dressings. Use wet towels in the stretcher if needed.

Transport or escort patient to sickbay 2-60-6-Q by the closest safe route.
Use Ammo elevators if possible to move injured personnel.
Are there any more medical emergencies ? Y/N

n

Are there any more emergencies or casualities ? Y/N

n

APPENDIX B

'DEMONSTRATION OF HANDLING A FIRE IN A BERTHING COMPARTMENT'

```
==============================================================
 This is a Demonstration Listing of the program Emergency
 handling a fire in a Berthing compartment that has no
 installed damage control equipment.
==============================================================

*************************************************************
*                  EMERGENCY                        *
* An Expert system to provide guidance on the operation of *
* installed damage control equipment aboard the ship.      *
* Answer the given questions and make the appropriate      *
* choices from the presented menus.                  *
*************************************************************


 Do you want to have the ship's compartments printed out ? Y/N

n

 Give the compartment  number of the location of the emergency .
 Enter the compartment number in format  __ ___ _-_
 Example : 5 60 0-E  ; NOTE:  The spaces are important.
If the exact location is unknown give the closest location known.
 The format must be accurate. For example  "1-Q " vice "1-q" .
3 30 2-L

The compartment submitted is  3- 30- 2-L

Is the nature of the emergency known ? Y/N

y
```

What is the nature of the emergency or problem ?
Enter the appropriate number below :

Fire = 1
Flooding = 2
Fumes = 3
Injury to personnel = 4

1
The following Berthing Compartments should be cleared :
3 - 30 -2-L

Set the following boundaries to contain the hazard :

Fitting No. WTH 2-38-1 located 2-37-1-Q
Fitting No. WTH 3-38-1 located 3-37-1-Q
Fitting No. WTH 3-38-2 located 3-30-4-L

Are you ready to continue with the program ? Y/N
y

To secure electrical power to compartment 3 - 30-2-L
Secure power at panel No. pan 2-53-2 located in 2-40-0-L FR 53 P

To secure ventilation to compartment 3 - 30-2-L
Use controller No. CONT 2-87-2 located in 2-80-4-Q FR 87 P

What type of fire is this ?
Enter the appropriate number below:

Class A (paper, wood, materials etc.) = 1
Class B (hydrocarbon or flammable liquids) = 2
Class C (electrical fire) = 3
Class D (Chemical fire) = 4
Type of fire unknown = 0

1
Is the source of the fire known? Y/N
n
Is the compartment number for the location of the emergency
correct as originally reported ? Y/N
y

Ensure fire boundaries are maintained.
Ensure that firefighters are properly equiped.
No appropriate installed systems, use nearest fire station.

Is the fire out ? Y/N

n

Continue to combat the fire until it is extinguished.

Is the fire out ? Y/N

y

Conduct the following tests:

Test for explosive gases .
Test for oxygen.
Test for carbon monoxide.
Test for carbon dioxide.

What type of fumes are present in the space ?
Enter the appropriate number.

Compartment Tested, Adequate Oxygen,
and no fumes Detected = 0
Explosive gases = 1
Hydrogen Sulfide = 2
Benzine = 3
Chlorine = 4
Carbon Monoxide = 5
Carbon Dioxide = 6
Unknown Fumes = 7

1

Put out the smoking lamp throughout the ship.
Set the reflash watch in OBAs and overhaul the fire.

Recommend use of portable pneumatic blowers to remove explosive vapors.
No installed exhaust ventilation systems.
Use a portable pneumatic blower or a safety checked Red Devil blower.

Is the fire overhauled ? Y/N

n

Continue to overhaul the fire . Conduct second
test when the overhaul is completed.
Conduct the following tests:

Test for explosive gases .
Test for oxygen.
Test for carbon monoxide.
Test for carbon dioxide.

What type of fumes are present in the space ?
Enter the appropriate number.

Compartment Tested, Adequate Oxygen,
and no fumes Detected = 0
Explosive gases = 1
Hydrogen Sulfide = 2
Benzine = 3
Chlorine = 4
Carbon Monoxide = 5
Carbon Dioxide = 6
Unknown Fumes = 7

0

Compartment 3- 30- 2-L is safe to enter .
No ventilation is needed except for Smoke removal
Does the compartment need to be desmoked ? Y/N

y

No installed exhaust ventilation systems.
Use a portable pneumatic blower or a safety checked Red Devil blower.

Does Compartment 3- 30- 2-L have a significant amount of
water, over two inches, that needs to be removed .

n

Recommend the use of mops and pails to clean up.
Use the closest deck drains to remove the water.

Are there any more emergencies or casualities ? Y/N

n

APPENDIX C

'DEMONSTRATION OF A FIRE IN A MAIN MACHINERY ROOM'

```
======================================================================
The following is a listing of the program Emergency handling
a complex fire casualty in a Main Machinery Room.  A listing
of the available compartments was made. This demonstration
shows the interaction between the program and the user.
======================================================================

****************************************************************
*                    EMERGENCY                    *
*  An Expert system to provide guidance on the operation of  *
*  installed damage control equipment aboard the ship.      *
*  Answer the given questions and make the appropriate      *
*  choices from the presented menus.                *
****************************************************************


Do you want to have the ship's compartments printed out ? Y/N
y

The following compartments will work as input for the program.

Compt number           Use              Installed systems

  01- 50- 3-L    Berthing Compartment  :  NONE
  02- 50- 1-Q    Passageway            :  NONE
  04- 40- 0-C    Control space         :  NONE

   2- 30- 2-Q    Workcenter            :  NONE
   2- 30- 4-Q    Passageway            :  NONE
   2- 30- 2-Q    Workcenter            :  Dry standpipe sprinkler
   2- 30- 1-Q    Workcenter            :  CO2 Hose Reel
   2- 30- 6-L    Head space            :  NONE
   2- 30- 5-Q    Admin space           :  NONE
   2- 30- 3-Q    Passageway            :  NONE
   2- 36- 2-Q    Repair Locker         :  NONE
   2- 37- 1-Q    Passageway            :  NONE
```

2- 40- 1-Q	Admin space	: NONE
2- 40- 2-Q	Galley space	: NONE
2- 40- 0-L	Galley space	: NONE
2- 48- 2-Q	Galley space	: Dry standpipe sprinkler
2- 50- 1-Q	Workcenter	: NONE
2- 60- 6-Q	Medical Spaces	: NONE
2- 60- 3-Q	Admin space	: NONE
2- 60- 2-Q	Passageway	: NONE
2- 60- 1-Q	Passageway	: NONE
2- 60- 0-L	Galley space	: NONE
2- 68- 2-L	Head space	: NONE
2- 70- 1-Q	Admin space	: NONE
2- 76- 1_Q	Passageway	: NONE
3- 30- 3-L	Head space	: NONE
3- 30- 2-L	Berthing Compartment	: NONE
3- 60- 7-L	Head space	: Installed exhaust fan
3- 70- 4-L	Head space	: Installed exhaust fan
3- 70- 2-L	Berthing Compartment	: NONE
3- 73- 1-Q	Passageway	: NONE
3- 80- 0-L	Berthing Compartment	: NONE
3- 90- 3-E	Cominuter space	: Installed eductor
3- 90- 3-Q	Workcenter	: Installed exhaust fan
4- 30- 1-A	Storeroom	: Dry standpipe sprinkler
4- 30- 2-M	AMMO Magazine	: Active sprinkler sys
4- 80- 1-Q	Passageway	: NONE
4- 80- 0-C	Control space	: NONE
4- 80- 6-Q	Passageway	: NONE
4- 80- 4-A	Storeroom	: Installed exhaust fan
5- 30- 1-A	Storeroom	: Dry standpipe sprinkler
5- 30- 4-M	AMMO Magazine	: Active sprinkler sys
5- 40- 0-E	Main Machinery Room	: HALON, TAU, and CO2 hosereel
5- 60- 0-E	Main Machinery Room	: HALON, TAU, and CO2 hosereel
6- 30- 1-A	Flammable storeroom	: CO2 Flooding
6- 30- 2-E	Pump room	: AFFF HR and CO2 hosereel
6- 30- 3-A	Storeroom	: Dry standpipe sprinkler
6- 80- 0-E	Pump room	: AFFF HR and CO2 hosereel
6- 90- 0-A	Hazardous materials	: CO2 Flooding

Give the compartment number of the location of the emergency .
Enter the compartment number in format __ ___ _-_
Example : 5 60 0-E ; NOTE: The spaces are important.
If the exact location is unknown give the closest location known.
The format must be accurate. For example "1-Q " vice "1-q" .
5 40 0-E

The compartment submitted is 5- 40- 0-E

Is the nature of the emergency known ? Y/N

y

What is the nature of the emergency or problem ?

Enter the appropriate number below :

Fire = 1

Flooding = 2

Fumes = 3

Injury to personnel = 4

1

Set the following boundaries to contain the hazard :

Fitting No. WTH 3-41-1 located 3-40-1-T
Fitting No. WTH 3-59-2 located 3-58-2-Q
Fitting No. WTS 3-59-1 located 3-58-1-T
Fitting No. WTH 2-41-1 located 2-40-1-Q
Fitting No. WTH 2-59-2 located 2-58-2-Q

Are you ready to continue with the program ? Y/N

y

To secure electrical power to compartment 5 - 40-0-E
Secure power at panel No. pan 2-23-1 located in 2-20-1-Q FR 23 P

To secure ventilation to compartment 5 - 40-0-E
Use controller No. CONT 2-59-2 located in *2-40-0-L FR 59 C

What type of fire is this ?

Enter the appropriate number below:

Class A (paper, wood, materials etc.) = 1
Class B (hydrocarbon or flammable liquids) = 2
Class C (electrical fire) = 3
Class D (Chemical fire) = 4
Type of fire unknown = 0

2

Is the source of the fire known? Y/N

n

Flammable liquids in some piping systems.

Is the compartment number for the location of the emergency correct as originally reported ? Y/N

y

Ensure fire boundaries are maintained.
Ensure that firefighters are properly equiped.
Oil spray fires can get out of control quickly.
Secure fuel source and localize fire as soon as
possible without endangering personnel.
Secure the exhaust ventilation to the space.
The compartment has an installed exhaust system.
Controller number cont 2-54-1 located 2-40-0-L FR 54 S

To operate the installed exhaust fan, open any closed
exhaust vent dampers and then activate the exhaust fan ,
at the given controller. In the event of a fire, secure
the fan at the given controller and then close the
exhaust vent dampers.

Are you ready to continue with the program ? Y/N
y
The compartment contains various firefighting systems.
Use the most appropriate system.
The compartment contains a Twin Agent Extinguishing System.
Valve number cont 5-45-1 located 5-40-0-E LL STBD
Valve number cont 4-53-2 located 5-40-0-E UL PORT
Valve number cont 4-57-1 located 5-40-0-E UL STBD

To operate the Twin Agent Fire Extinguishing System , open
the given control valve .
To secure the TAFES system follow the procedures given in
NAVSEA Tech Manual Chapter 9930, Firefighting-Ship.

Are you ready to continue with the program ? Y/N
y
The compartment contains a AFFF sprinkler system.
Valve number val 4-42-2 located 4-40-0-C FR 42 P
Valve number val 2-54-2 located 2-40-0-L FR 54 P

To operate the AFFF sprinkler system, operate
given fitting.

Are you ready to continue with the program ? Y/N
y
The compartment contains a AFFF Hose Reel system.

To operate the AFFF Hose Reel System , open
the given control valve .
In most cases a stream of foam should be deflected from
a bulkhead to avoid agitation of burning liquids. Foam
is only effective when it covers the top of the surface
of burning combustibles.
To secure the Hose Reel follow the procedures given in
NAVSEA Tech Manual Chapter 9930, Firefighting-Ship.

Are you ready to continue with the program ? Y/N
y
The compartment's HALON actuators are as follows :
Actuator act# 4-40-2 located 4-40-0-C FR 40 P
Actuator act# 2-40-1 located 2-40-1-Q FR 40 P

Before operating the installed HALON system , all supply
and exhaust vent dampers should be closed if possible,
and all fans serving the space secured. All personnel
should be out of the space or in the process of leaving
the space. To activate the HALON system use the given
local or remote pneumatic actuators.

Are you ready to continue with the program ? Y/N
y
Activate AFFF Bilge sprinkler system.
The compartment contains a AFFF sprinkler system.
Valve number val 4-42-2 located 4-40-0-C FR 42 P
Valve number val 2-54-2 located 2-40-0-L FR 54 P

To operate the AFFF sprinkler system, operate
given fitting.

Are you ready to continue with the program ? Y/N
y

Is the fire out ? Y/N

y

Conduct the following tests:

Test for explosive gases .
Test for oxygen.
Test for carbon monoxide.
Test for carbon dioxide.

What type of fumes are present in the space ?
Enter the appropriate number.

Compartment Tested, Adequate Oxygen,
and no fumes Detected = 0
Explosive gases = 1
Hydrogen Sulfide = 2
Benzine = 3
Chlorine = 4
Carbon Monoxide = 5
Carbon Dioxide = 6
Unknown Fumes = 7

5

Carbon Monoxide can be deadly. All personnel who might be exposed in the course of finding and elimating the source of the fumes should use breathing devices.
Set the reflash watch in OBAs and overhaul the fire.

The compartment has an installed exhaust system.
Controller number cont 2-54-1 located 2-40-0-L FR 54 S

To operate the installed exhaust fan, open any closed exhaust vent dampers and then activate the exhaust fan , at the given controller. In the event of a fire, secure the fan at the given controller and then close the exhaust vent dampers.

Are you ready to continue with the program ? Y/N
y

Is the fire overhauled ? Y/N

y

Conduct the following tests:

Test for explosive gases .
Test for oxygen.
Test for carbon monoxide.
Test for carbon dioxide.

What type of fumes are present in the space ?
Enter the appropriate number.

Compartment Tested, Adequate Oxygen,
and no fumes Detected = 0
Explosive gases = 1
Hydrogen Sulfide = 2
Benzine = 3
Chlorine = 4
Carbon Monoxide = 5
Carbon Dioxide = 6
Unknown Fumes = 7

0

Compartment 5- 40- 0-E is safe to enter .
No ventilation is needed except for Smoke removal
Does the compartment need to be desmoked ? Y/N

y

The compartment has an installed exhaust system.
Controller number cont 2-54-1 located 2-40-0-L FR 54 S

To operate the installed exhaust fan, open any closed exhaust vent dampers and then activate the exhaust fan , at the given controller. In the event of a fire, secure the fan at the given controller and then close the exhaust vent dampers.

Are you ready to continue with the program ? Y/N
y

The main circ pump can be used to dewater a Main Machinery space.
The compartment can be dewatered using the main drainage system.
Valve number OVBd 5-43-1 located 5-40-0-E FR 43 S
Valve number ActW 5-44-1 located 5-40-0-E FR 45 S
Valve number Suct 5-42-1 located 5-40-0-E FR 42 S
Valve number OVBd 5-54-2 located 5-40-0-E FR 54 P
Valve number ActW 5-55-4 located 5-40-0-E FR 55 P
Valve number Suct 5-58-2 located 5-40-0-E FR 58 P

To operate the installed eductor system, open the (OVBD)
overboard valve first, then the activating water (ACTW),
and finally open the bilge or main drain (SUCT) valve.
To secure the eductor use the reverse order of lineup.

Are you ready to continue with the program ? Y/N
n
Answer yes when you are ready to continue.
Are you ready to continue with the program ? Y/N
y

Are there any more emergencies or casualities ? Y/N

n
Execution terminated.

25179 statements executed in 0004.700 seconds cpu time.

APPENDIX D

LISTING OF PROGRAM EMERGENCY

```
program emergency(input, output);

{*****************************************************************
{*  TITLE :      EMERGENCY
{*  AUTHOR      : LCDR Bernard G Gogel
{*  Date Written : 21 Sept 86 - 16 Nov 86
{*
{*  Version      :  One
{*  System Used  :  Turbo Pascal on COMPAQ  Deskpro  PC
{*
{*  I/O Sources  :  Data input from disk files and user input
{*                 from the terminal Keyboard.
{*  Description  :  This program is an expert system to provide
{*                 guidance in the operation of installed damage
{*                 control systems aboard Naval ships.
{*****************************************************************

                    (*   GLOBAL  CONSTANTS    *)
const
   numdecks  = 6  ;   (* Number of decks on the ship *)
   numlevel  = 4  ;  (* Number of levels on the ship *)
   numcompt  =  100 ;  (* Number of compartments on the ship *)
   numfittings =  50 ; (* Number of fittings on the ship *)
   numelpanel =  28 ;  (* Number of electrical panels  *)
   numventcon  =  30 ;  (* Number of ventilation controllers *)
   numhalon  = 4  ;   (* Number of HALON systems actuators on the
                          ship *)
   numcofld   = 6  ;  (* Number of CO2 flooding systems
                          controllers *)
   numcohr   =  7 ;    (* Number of CO2 hose reels on the ship *)
   numafffsp =  4 ;   (* Number of AFFF sprinkler systems
                          controllers *)
   numafffhr =  4 ;   (* Number of AFFF hose reels controls on the
                          ship  *)
   numtau = 8 ;       (* Number of Twin Agent Units on the ship
   numeduct = 18 ;    (* Number of installed eductor system valves
   numexhfn  =  13 ;   (* Number of installed exhausr fans on the
                          ship *)
   numactsp   = 8  ;  (* Number of active sprinkler systems on the
                          ship *)
   numdysp = 12  ;    (* Number of dry standpipe sprinklers on the
                          ship *)
   sickbaydk = 2  ;  (* Sickbay on the second deck *)
   sickbayfr = 60  ; (* Sickbay at frame 60       *)
   sickbay = '2-60-6-Q' ; (* compartment number *)


type
   emertypes = ( fire, flooding , fumes, personinj ) ;

   filname = string[16] ;
   filvar = text ;

   recordptr = @compartmentrec  ;
   usearray  =  packed array (.1..4.) of char ;
   fncarray  =  packed array (.1..13.) of char ;
   locarray  =  packed array (.1..21.) of char ;
```

```
compartmentrec = record
     dklevel       : integer ;
     frame         : integer ;
     compabmndk    : boolean ;
     loctocl       : usearray; (* location to centerline of ship
     shipside      : char    ; (* Port, Stbd, or Centerline *)
     zone          : integer ; (* zone for setting boundaries *)
     panelcode     : integer ; (* code number for electric panel *)
     ventcode      : integer ; (* code number for ventilation controller *)
     pipingcode    : integer ; (* code number for piping in the space *)
     usecode       : integer ; (* code number for use of space *)
     instalsystems : integer ;
     linkit        : recordptr ;
end ;  (* compartmentrec  *)

fittings = record
     zoneloc  : integer ;    (* zone where fitting is located *)
     fitting  : fncarray ;   (* fitting number *)
     fitlocat : locarray ;   (* location of fitting *)
end ;  (* fittings *)

elecpanel = record
     panlcode  : integer ;   (* code for electrical panel  *)
     panelnum  : fncarray ;  (* electrical panel number *)
     panlocat  : locarray ;  (* electrical panel location  *)
end ;  (* elecpanel *)

ventcontrl = record
     vntcode   : integer  ;  (* code for ventilation controller  *)
     contnum   : fncarray ;  (* number of ventilation controller *)
     contlocat : locarray ; (* location of ventilation controller *)
end ;  (* ventcontrl *)

instlsys = record
     dkorlev   : integer ;   (* deck or level of compartment *)
     fr        : integer ;   (* frame of compartment  *)
     thirdpt   : usearray ;  (* third part of compartment number *)
     valvenum  : fncarray ;  (* valve or system control number *)
     location  : locarray ;  (* location of controls *)
end ; (* instlsys *)

upprcomp = array (.1..numlevel.) of compartmentrec ;
lowrcomp = array (.1..numdecks.) of compartmentrec ;
closures = array (.1..numfittings.) of fittings ;
panels  = array (.1..numelpanel.) of elecpanel ;
ventcntls = array (.1..numventcon.) of ventcontrl ;
halonsys = array (.1..numhalon.) of instlsys ;
co2flood = array (.1..numcofld.) of instlsys ;
co2hoser = array (.1..numcohr.) of instlsys ;
afffsprk = array (.1..numafffsp.) of instlsys ;
afffhsrl = array (.1..numafffhr.) of instlsys ;
tausystm = array (.1..numtau.) of instlsys ;
insteduc = array (.1..numeduct.) of instlsys ;
exhstfans = array (.1..numexhfn.) of instlsys ;
actsprsy = array (.1..numactsp.) of instlsys ;
drystand = array (.1..numdysp.) of instlsys ;

                                   (* GLOBAL VARIABLES *)
var

   compt       : compartmentrec ;
   uplev       : upprcomp ;
   lwrdk       : lowrcomp ;
   closure     : closures ;
   panel       : panels   ;
   vent        : ventcntls ;
   halon       : halonsys ;
   co2fld      : co2flood ;
   co2hr       : co2hoser ;
```

```
    afffspr      : afffsprk ;
    afffhr       : afffhsrl ;
    tausys       : tausystm ;
    eductor      : insteduc ;
    exhfan       : exhstfans ;
    actspr       : actsprsy ;
    drystd       : drystand ;

    answer       : char ;
    emerover     : boolean ;  (* used to determine when the emergency is over*)
    abmndeck     : boolean ;  (* Used to determine if comp above or below mn dk *)
    haztype      : emertypes; (* used to determine the type of hazard  *)
    newrecord, nextrec, lastrec, needrec : recordptr ;   (* Pointer Variables *)

(****************************************************************************)
(*                    Procedure Continue                            *)
(* This procedure is used to provide a pause for the user to read the   *)
(* information on the terminal screen before displaying the next menu   *)
(* or informational listing. The user enter a "Y" to continue.          *)
(****************************************************************************)
procedure continue ;

var
     ans      : char;
     contprog : boolean  ;  (* used to control loop *)

begin
     contprog := false ;
     writeln;
     repeat
         writeln('Are you ready to continue with the program ? Y/N ');
         readln(ans);
         if (ans = 'Y') or ( ans = 'y') then
              contprog := true
         else
              writeln(' Answer yes when you are ready to continue.');
     until contprog ;
end ; (* continue *)

(*****************************************************************************)
(*                       Procedure Printout                            *)
(*  This procedure prints out information on the compartment pointed to by  *)
(*  the passed parameter. This procedure is called by procedure printcomp.  *)
(*****************************************************************************)

procedure printout( temprec : recordptr) ;

begin
     if temprec@.compabmndk then
          write('0') ;
     write( temprec@.dklevel , '-',temprec@.frame:3,'-',temprec@.loctocl:4,
                                                        );

     case temprec@.usecode of
         1 : write(' Admin space            : ');
         2 : write(' Berthing Compartment  : ');
         3 : write(' Passageway            : ');
         4 : write(' Head space            : ');
         5 : write(' Workcenter            : ');
         6 : write(' Main Machinery Room   : ');
         7 : write(' Boiler Room           : ');
         8 : write(' Pump room             : ') ;
         9 : write(' Emergency Generator   : ') ;
        10 : write(' Cominuter space       : ') ;
        11 : write(' Galley space          : ') ;
        12 : write(' Storeroom             : ') ;
        13 : write(' Flammable storeroom   : ') ;
        14 : write(' Fuel tank             : ') ;
        15 : write(' Water tank            : ') ;
```

```
      16 : write(' Void                  : ') ;
      17 : write(' Access Trunk          : ') ;
      18 : write(' Control space         : ') ;
      19 : write(' Fan Room              : ') ;
      20 : write(' Repair Locker         : ') ;
      21 : write(' Hazardous materials   : ');
      22 : write(' AMMO Magazine         : ');
      23 : write(' Medical Spaces        : ');
    end ;  (* case *)

    case temprec@.instalsystems of
       0 : writeln(' NONE ');
       1 : writeln(' Installed eductor');
       2 : writeln(' Installed exhaust fan ');
       3 : writeln(' Active sprinkler sys');
       4 : writeln(' Dry standpipe sprinkler ');
       5 : writeln(' CO2 Flooding ');
       6 : writeln(' CO2 Hose Reel ');
       7 : writeln(' HALON ');
       8 : writeln(' Twin Agent Unit ');
       9 : writeln(' AFFF Sprinkler ');
      10 : writeln(' AFFF hose Reel ');
      11 : writeln(' HALON, TAU, and CO2 hosereel ');
      12 : writeln(' AFFF HR and CO2 hosereel ');
    end ;  (* case *)
end ;   (* printout *)

(****************************************************************************)
(*          Procedure (printcomp) Print Compartments                        *)
(*  This procedure prints out a listing of all the ship's compartments      *)
(*  that are in the data base. The procedure calls procedure printout      *)
(*  to print information on a given compartment.                            *)
(****************************************************************************)

procedure printcomp ;

var
   i : integer  ;


begin
    writeln;
    writeln(' Compt number            Use            Installed systems');
    writeln;
    for i := 1 to numlevel do
        begin
            nextrec := uplev(.i.).linkit ;
            while nextrec <> nil do
                begin
                    printout(nextrec);
                    nextrec := nextrec                    .linkit ;
                end ; (* while *)
        end ;  (* for *)
    writeln;
    for i := 1 to numdecks do
        begin
            nextrec := lwrdk(.i.).linkit ;
            while nextrec <> nil do
                begin
                    printout(nextrec);
                    nextrec := nextrec                    .linkit ;
                end ; (* while *)
        end ;  (* for *)
    writeln ;
end ;  (* procedure printcomp *)
```

```
(**************************************************************************)
(*                  Procedure Insertrec                          *)
(* This procedure is called by procedure  loadfiles to insert a record *)
(* in an array of linked list of records.                         *)
(* This procedure contains procedure midinsert to insert a record in   *)
(* the middle of the linked list of records.                       *)
(**************************************************************************)

procedure insertrec( updw : char; dk, fr : integer ) ;

(**--------------------Procedure  Midinsert-------------------------**)

procedure midinsert ;

var
   recinsert : boolean ; (* used to determine if the record was inserted *)

begin
     nextrec := lastrec@.linkit ;
     if nextrec = nil then
         begin
              lastrec@.linkit := newrecord ;
              lastrec := newrecord ;
         end   (* If *)
     else   (* outer else *)
         begin
              recinsert := false ;
              while (nextrec < > nil) and (not recinsert) do
                  begin
                       if nextrec@.frame > = fr then
                           begin
                           newrecord@.linkit := lastrec          .linkit ;
                                lastrec@.linkit := newrecord;
                                recinsert := true;
                           end  (* If *)
                       else
                           begin
                           lastrec := nextrec ;
                                nextrec := nextrec@.linkit ;
                           end; (* else *)
                       if (nextrec = nil) and (not recinsert ) then
                           begin
                           lastrec@.linkit := newrecord ;
                                recinsert := true ;
                           end ; (* if *)
                  end; (* while *)
         end;     (* outer else *)
end ;  (* midinsert  *)

                                   (* Start of insertrec *)
begin
     if updw = 'T' then
         begin
            if uplev(.dk.).linkit = nil then
                  uplev(.dk.).linkit := newrecord
            else
                  begin
                      lastrec := uplev(.dk.).linkit ;
                      if lastrec@.frame > fr then
                           begin
                                newrecord@.linkit := lastrec;
                                uplev(.dk.).linkit := newrecord ;
                                nextrec := lastrec ;
                                lastrec := newrecord ;
                           end  (* if *)
                      else
                                midinsert ;
                  end ;  (* else *)
         end ;  (* if T  *)
```

```
    if updw =-'F' then
        begin
           if lwrdk(.dk.).linkit = nil then
                 lwrdk(.dk.).linkit := newrecord
           else
                 begin
                    lastrec := lwrdk(.dk.).linkit ;
                    if lastrec@.frame > fr then
                        begin
                              newrecord@.linkit := lastrec;
                              lwrdk(.dk.).linkit := newrecord ;
                              nextrec := lastrec ;
                              lastrec := newrecord ;
                        end (* if *)
                    else
                          midinsert ;
                 end ; (* else *)
        end ; (* if F   *)
end ; (* procedure insertrec *)

(**************************************************************************)
(*                     Procedure Loadfiles                        *)
(*   This procedure reads in the coded information on all the ship's    *)
(* compartments from a disk file into a linked list of records. There   *)
(* is one record per compartment on the ship. Loadfiles calls procedure *)
(* Insertrec to insert the records in the proper position in an array   *)
(* of linked list of records.                                     *)
(**************************************************************************)
procedure loadfiles ;

var
   i ,                          (* used as an index  *)
   gdk, gfr, gzone, gpanc, gvenc,
   gpipngc, ucode, getint : integer ;    (* used to read in data from file *)
   abmndk, side, discard : char ;        (* used to read in data from file *)
   rloc : usearray ;                   (* used to read in data from file *)
   infile : filvar ;


begin
                        (* Initialize pointers to Nil *)
    for i := 1 to numlevel do
        begin
           with uplev(.i.) do
                begin
                    linkit := nil;
                    dklevel := i ;
                end ; (* with *)
        end ; (* for *)
    for i := 1 to numdecks do
        begin
           with lwrdk(.i.) do
                begin
                    linkit := nil;
                    dklevel := i ;
                end ; (* with *)
        end ; (* for *)

                        (* Prepare for input *)
    assign(infile, 'compdata.txt ') ;
    reset(infile);
                           (* Read in information from file  *)
   while not eof (infile) do
      begin
          with compt do
              begin
                  new(newrecord);
                  read(infile, abmndk);
```

```
                readln( infile, gdk, gfr,discard, rloc,discard,side,
                            gzone, gpanc, gvenc, gpipngc, ucode, getint ) ;
                newrecord@.dklevel := gdk ;
                newrecord@.frame := gfr ;
                newrecord@.loctocl := rloc ;
                newrecord@.shipside := side ;
                newrecord@.zone := gzone ;
                newrecord@.panelcode := gpanc ;
                newrecord@.ventcode := gvenc ;
                newrecord@.pipingcode := gpipngc ;
                newrecord@.usecode := ucode ;
                newrecord@.instalsystems := getint ;
                if abmndk = 'F' then
                   newrecord@.compabmndk := false ;
                if abmndk = 'T' then
                   newrecord       .compabmndk := true ;
                newrecord@.linkit := nil ;
                insertrec( abmndk, gdk, gfr);
             end ;  (* with *)
        end ;  (* while *)
   close (infile ) ;

end ;  (*  loadfiles *)

(**********************************************************************)
(*                  Procedure Loaddata                        *)
(*   This procedure reads in the data concerning the ship's installed *)
(* damage control systems from a disk file and stores the information *)
(* in an array of records for each system.                     *)
(**********************************************************************)
procedure loaddata ;

var
   i       : integer ;           (* used as an index  *)
   discard : char ;  (* used to read in and discard seperators *)
   infile  : filvar ;

begin         (* Prepare for input *)
    assign( infile, 'electpan.txt');
    reset(infile);
                              (* Read in data from file *)
    i := 1  ;  (* initialize to 1  *)
    while not eof (infile) do
      begin
         with panel(.i.) do
              begin
                 readln(infile, panlcode, discard, discard, panelnum, discard,
                              discard, panlocat);
                 i := i + 1 ;
              end;  (* with *)
      end;  (* while *)
    close(infile);

                                (* Prepare for Input *)
    assign( infile, 'ventcont.txt');
    reset(infile);
                              (* Read in data from file *)
    i := 1  ;  (* initialize to 1  *)
    while not eof (infile) do
      begin
         with vent(.i.) do
              begin
                 readln(infile, vntcode, discard, discard, contnum, discard,
                           discard,  discard, contlocat);
                 i := i + 1 ;
              end;  (* with *)
      end;  (* while *)
    close(infile);
```

```
                          (* Prepare for Input *)
assign( infile, 'zonclose.txt');
reset(infile);
                              (* Read in data from file *)
i := 1  ;  (* initialize to 1  *)
while not eof (infile) do
  begin
     with closure(.i.) do
         begin
            readln(infile, zoneloc, fitting, fitlocat );
            i := i + 1 ;
         end;  (* with *)
  end;  (* while *)
close(infile);

                               (* Prepare for Input *)
assign( infile, 'actspk.txt');
reset(infile);
                              (* Read in data from file *)
i := 1  ;  (* initialize to 1  *)
while not eof (infile) do
  begin
     with actspr(.i.) do
         begin
            readln(infile, dkorlev, fr, discard, thirdpt, discard,
                          valvenum, discard, discard, location);
            i := i + 1 ;
         end;  (* with *)
  end;  (* while *)
close(infile);

                            (* Prepare for Input *)
assign( infile, 'afffhr.txt');
reset(infile);
                              (* Read in data from file *)
i := 1  ;  (* initialize to 1  *)
while not eof (infile) do
  begin
     with afffhr(.i.) do
         begin
            readln(infile, dkorlev, fr, discard, thirdpt, discard,
                          valvenum, discard, discard, location);
            i := i + 1 ;
         end;  (* with *)
  end;  (* while *)
close(infile);

                            (* Prepare for Input *)
assign( infile, 'afffspk.txt');
reset(infile);
                              (* Read in data from file *)
i := 1  ;  (* initialize to 1  *)
while not eof (infile) do
  begin
     with afffspr(.i.) do
         begin
            readln(infile, dkorlev,  fr, discard, thirdpt, discard,
                          valvenum, discard, discard, location);
            i := i + 1 ;
         end;  (* with *)
  end;  (* while *)
close(infile);

                            (* Prepare for Input *)
assign( infile, 'co2flod.txt');
reset(infile);
                              (* Read in data from file *)
i := 1  ;  (* initialize to 1  *)
while not eof (infile) do
```

```
  begin
     with co2fld(.i.) do
         begin
            readln(infile, dkorlev, fr, discard, thirdpt, discard,
                          valvenum, discard, discard, location);
            i := i + 1 ;
         end;  (* with *)
  end;  (* while *)
close(infile);
                           (* Prepare for Input *)
assign( infile, 'co2hrfl.txt');
reset(infile);
                             (* Read in data from file *)
i :=  1  ;  (* initialize to 1  *)
while not eof (infile) do
  begin
     with co2hr(.i.) do
         begin
            readln(infile, dkorlev,  fr, discard, thirdpt, discard,
                          valvenum, discard, discard, location);
            i := i + 1 ;
         end;  (* with *)
  end;  (* while *)
close(infile);
{.PA}

                           (* Prepare for Input *)
assign( infile, 'drystd.txt');
reset(infile);
                             (* Read in data from file *)
i :=  1  ;  (* initialize to 1  *)
while not eof (infile) do
  begin
     with drystd(.i.) do
         begin
            readln(infile, dkorlev, fr, discard, thirdpt, discard,
                          valvenum, discard, discard, location);
            i := i + 1 ;
         end;  (* with *)
  end;  (* while *)
close(infile);

                           (* Prepare for Input *)
assign( infile, 'exhfan.txt');
reset(infile);
                             (* Read in data from file *)
i :=  1  ;  (* initialize to 1  *)
while not eof (infile) do
  begin
     with exhfan(.i.) do
         begin
            readln(infile, dkorlev, fr, discard, thirdpt, discard,
                          valvenum, discard, discard, location);
            i := i + 1 ;
         end;  (* with *)
  end;  (* while *)
close(infile);

                           (* Prepare for Input *)
assign( infile, 'halonfil.txt');
reset(infile);
                             (* Read in data from file *)
i :=  1  ;  (* initialize to 1  *)
while not eof (infile) do
  begin
     with halon(.i.) do
         begin
            readln(infile, dkorlev, fr, discard, thirdpt, discard,
                          valvenum, discard, discard, location);
```

```
                i := i + 1 ;
            end;  (* with *)
        end;  (* while *)
    close(infile);

                        (* Prepare for Input *)
    assign( infile, 'ineduct.txt');
    reset(infile);
                         (* Read in data from file *)
    i := 1  ;  (* initialize to 1  *)
    while not eof (infile) do
      begin
         with eductor(.i.) do
            begin
               readln(infile, dkorlev, fr, discard, thirdpt, discard,
                           valvenum, discard, discard, location);
               i := i + 1 ;
            end;  (* with *)
      end;  (* while *)
    close(infile);

                        (* Prepare for Input *)
    assign( infile, 'tafesfil.txt');
    reset(infile);
                         (* Read in data from file *)
    i := 1  ;  (* initialize to 1  *)
    while not eof (infile) do
      begin
         with tausys(.i.) do
            begin
               readln(infile, dkorlev, fr, discard, thirdpt, discard,
                           valvenum, discard, discard, location);
               i := i + 1 ;
            end;  (* with *)
      end;  (* while *)
    close(infile);

end;   (* loaddata *)


(*************************************************************************)
(*                  Procedure getcompnum                          *)
(* This procedure requests and reads in the compartment number of the  *)
(* space where the emergency has occurred. The record of the submitted *)
(* compartment is found and pointed to by the pointer variable needrec.*)
(*************************************************************************)
procedure getcompnum( var needrec : recordptr );

var
   getch1, getch2, getch3  : char  ;
   rdklevel,                 (* used to read in reported deck level *)
   rdframe   : integer ;    (* used to read in frame number  *)
   cfound    : boolean ;     (* used to determine if comp found *)
   readloc   : usearray ;   (* used to read in use of comp *)

begin
    cfound := false;
    repeat;
       writeln;
       writeln(' Give the compartment number of the location of the emergency .');
       writeln(' Enter the compartment number in format __ ___ _-_ ');
       writeln(' Example :  5 60 0-E   ;  NOTE:   The spaces are important. ');
       writeln('If the exact location is unknown give the closest location known.');
       writeln(' The format must be accurate. For example  "1-Q " vice  "1-q" .');
       readln(getch1, getch2, rdframe, getch3, readloc) ;
       writeln;
       if getch1  =  '0' then
          begin
             abmndeck := true ;
```

```
      case getch2 of
        '1' : rdklevel := 1 ;
        '2' : rdklevel := 2 ;
        '3' : rdklevel := 3 ;
        '4' : rdklevel := 4 ;
        '5' : rdklevel := 5 ;
        '6' : rdklevel := 6 ;
        '7' : rdklevel := 7 ;
        '8' : rdklevel := 8 ;
        '9' : rdklevel := 9 ;
      end ; (* case *)
    end  (* if *)
else
    begin
      abmndeck := false ;
      case getch1 of
        '1' : rdklevel := 1 ;
        '2' : rdklevel := 2 ;
        '3' : rdklevel := 3 ;
        '4' : rdklevel := 4 ;
        '5' : rdklevel := 5 ;
        '6' : rdklevel := 6 ;
        '7' : rdklevel := 7 ;
        '8' : rdklevel := 8 ;
        '9' : rdklevel := 9 ;
      end ; (* case *)
    end;  (* else *)

if abmndeck then
     writeln('The compartment number submitted is  0', rdklevel,'-',
                  rdframe, '-',readloc)
else
    writeln('The compartment number submitted is ', rdklevel,'-',
                  rdframe, '- ',readloc);
  if abmndeck then
     begin
          nextrec := uplev(.rdklevel.).linkit ;
          while  (nextrec < > nil ) and (not cfound ) do
               begin
                    if ( nextrec@.frame  =  rdframe) and (nextrec@.loctocl  =  readloc) then
                         begin
                              needrec := nextrec ;
                              cfound := true ;
                         end   (* if *)
                    else
                         nextrec := nextrec@.linkit ;
               end ;  (* while *)
          if not cfound then
          writeln(' Compartment number is not in Ship°s data file. Check ',
                              'compartment number.');
     end   (* if *)

  else
     begin
          nextrec := lwrdk(.rdklevel.).linkit ;
          while  (nextrec < > nil ) and (not cfound ) do
               begin
                    if ( nextrec@.frame  =  rdframe) and (nextrec@.loctocl  =  readloc) then
                         begin
                              needrec := nextrec ;
                              cfound := true ;
                         end   (* if *)
                    else
                         begin
                           nextrec := nextrec@.linkit ;
                         end;
               end ;  (* while *)
          if not cfound then
          writeln(' Compartment number is not in Ship°s data file. Check ',
```

```
                                        'compartment number.');
            end;  (* else *)
      until cfound ;

end ;  (* procedure getcompnum *)


(*****************************************************************************)
(*                  Procedure (unkwnature) Unknown Nature                    *)
(* This procedure is called when the nature of the emergency is not         *)
(* known . The procedure provides guidance to the user on the methods       *)
(* that can be used to determine the nature of the emergency.               *)
(*****************************************************************************)
procedure unkwnature(needrec : recordptr) ;

begin
      writeln('Send investigators to the area to investigate.  ') ;
      if needrec@.usecode = 21 then
         writeln(' Hazardous materials in space. Possible source of fire',
                        ' or fumes.');
      if needrec@.usecode = 13 then
         writeln(' Flammable Storeroom. High Fire Danger. ');
      if (needrec@.pipingcode > 5) and (needrec@.pipingcode < 7) then
         writeln(' Possible toxic gases from plumbing system. ');
      writeln(' This procedure will be developed futher in the future.');
      writeln;
end;  (* procedure unkwnature *)

(*****************************************************************************)
(*                   Procedure (natknown) Nature Known                        *)
(*  This procedure is called when the nature of the emergency is known.      *)
(*  The procedure calls perinjury in the event of an injury to personnel     *)
(*  and calls procedures boundaries, idensource, elimhazard, recover, in     *)
(*  the event of fire, flooding, or fumes .                                  *)
(*  This procedure contains procedures fumetest, safeprec, perinjury,        *)
(*  boundaries, idensource, sysguide, elimhazard, recover, and ten           *)
(*  similiar Get System information Procedures.                              *)
(*****************************************************************************)

procedure natknown(needrec : recordptr)  ;

var
      hazcode  : integer ;  (* used to identify the type of hazard *)
      hazsource : integer ;

(*****************************************************************************)
(*                   Procedure Fume Test  ( fumetest)                        *)
(* This procedure provides guidance on the proper tests to conduct after    *)
(* a fire is extinguished or when fumes are discovered in a compartment.    *)
(* Fumetest is called by procedures idensource and recover.                 *)
(*****************************************************************************)
procedure fumetest( needrec: recordptr ; haztype : emertypes;
                                    var fumtype : integer );
var
      fumefnd  : boolean ;   (* used to determine if nature of fumes found *)

begin
      fumefnd := false;
      if haztype = fire then
         begin
            writeln(' Conduct the following tests: ' );
            writeln;
            writeln(' Test for explosive gases .');
            writeln(' Test for oxygen. ' );
            writeln(' Test for carbon monoxide.');
            writeln(' Test for carbon dioxide.');
            writeln;
         end; (* if haztype fire  *)
```

```
    if haztype = fumes then
       begin
          writeln(' Conduct the following tests: ' );
          writeln;
          writeln(' Test for explosive gases .');
          writeln(' Test for oxygen. ' );
          writeln(' Test for carbon monoxide.');
          writeln(' Test for carbon dioxide.');
          if needrec@.usecode = 10 then
             writeln(' Test for hydrogen sulfide. ');
          if (needrec@.usecode < 9) and (needrec .usecode > 5) then
             writeln(' Test for chlorine ');
          if needrec@.usecode = 21 then
             writeln(' Test for all hazrdous gases. ');
       end; (* if haztype fumes  *)

    repeat
       writeln(' What type of fumes are present in the space ? ');
       writeln(' Enter the appropriate number. ');
       writeln;
       writeln(' Compartment Tested, Adequate Oxygen, ');
       writeln(' and no fumes Detected        = 0 ' );
       writeln('  Explosive gases             = 1 ' );
       writeln('  Hydrogen Sulfide            = 2 ' );
       writeln('  Benzine                     = 3 ' );
       writeln('  Chlorine                    = 4 ' );
       writeln('  Carbon Monoxide             = 5 ' );
       writeln('  Carbon Dioxide              = 6 ' );
       writeln('  Unknown Fumes               = 7 ' );
       writeln;
       readln(fumtype);
       if (fumtype > 7 ) or (fumtype < 0 ) then
             writeln(' Need a valid input, please try again. ')
       else
             fumefnd := true ;
    until fumefnd  ;
end ; (* fumetest *)

(*****************************************************************************)
(*              Procedure (safeprec) Safety Precautions                     *)
(* This procedure provides safety precautions to the user for the type      *)
(* of hazard and it's source that are passed to the procedure.              *)
(* Safeprec is called by procedures idensource and elimhazard.              *)
(*****************************************************************************)

procedure safeprec( haztype : emertypes; hazsource : integer );

begin
    writeln;
    if haztype = fire then
       begin
          case hazsource of
            1 : begin       (* Class A *)
                   writeln(' Ensure fire boundaries are maintained.');
                   writeln(' Ensure that firefighters are properly equiped.');
                end; (* 1 *)

            2 : begin       (* Class B *)
                   writeln(' Ensure fire boundaries are maintained.');
                   writeln(' Ensure that firefighters are properly equiped.');
                   writeln(' Oil spray fires can get out of control quickly.');
                   writeln(' Secure fuel source and localize fire as soon as');
                   writeln(' possible without endangering personnel.');
                end; (* 2 *)

            3 : begin       (* Class C *)
                   writeln(' Ensure electrical power is secured.');
                   writeln(' Ensure fire boundaries are maintained.');
                   writeln(' Ensure that firefighters are properly equiped.');
```

```
            end; (* 3 *)

        4 : begin      (* Class D *)
                writeln(' Ensure fire boundaries are maintained.');
                writeln(' Ensure that firefighters are properly equiped to ');
                writeln(' handle all types of fires. It is extremely inportant');
                writeln(' to determine the nature and localize the chemical');
                writeln(' involved. Some materials react violently with water.');
                writeln(' Some metal fires are extremely difficult to extinguish.');
                writeln(' Metals such as magnesium, sodium, titanium,zinc, ');
                writeln(' zirconium, and potassium are class "D" combustibles.');
                writeln(' Refer to NAVSEA Tech Manual  Chapter 9930 ,');
                writeln('    FIREFIGHTING-SHIP for further information. ');
            end; (* 4 *)
         end ; (* case *)
     end; (* if haztype fire  *)

  if haztype = flooding then
       begin
         case hazsource of

     1,2,3 : writeln('Ensure electrical power to the space is secured.');

        4 : begin
                writeln('Put out the smoking lamp throughout the ship.');
                writeln('Ensure electrical power to the space is secured.');
                writeln('Secure the source of fuel as soon as possible.');
                writeln('Use non-sparking tools when working on the fuel system.');
            end;
         end;  (*  case *)
       end; (* if haztype flooding  *)

     if haztype = fumes then
       begin
         case hazsource of
          1 : writeln('Put out the smoking lamp throughout the ship.');

        2 : begin
                writeln(' Hydrogen Sulfide is highly toxic. All personnel who');
                writeln(' might be exposed in the course of finding and elimating');
                writeln(' the source of the fumes should use breathing devices.' );
            end ;

        3 : begin
                writeln(' Benzine can be harmful to people. All personnel who');
                writeln(' might be exposed in the course of finding and elimating');
                writeln(' the source of the fumes should use breathing devices.' );
            end ;

        4 : begin
                writeln(' Chlorine gas can be deadly. All personnel who');
                writeln(' might be exposed in the course of finding and elimating');
                writeln(' the source of the fumes should use breathing devices.' );
            end ;

        5 : begin
                writeln(' Carbon Monoxide can be deadly. All personnel who');
                writeln(' might be exposed in the course of finding and elimating');
                writeln(' the source of the fumes should use breathing devices.' );
            end ;

         6 : begin
                writeln(' Carbon Dioxide in large doses is toxic. All personnel who');
                writeln(' might be exposed in the course of finding and elimating');
                writeln(' the source of the fumes should use breathing devices.' );
            end ;
        end;  (* case *)
      end; (* if haztype fumes  *)
end ;  (* safeprec *)
```

```
(******************************************************************************)
(*                Procedure ( perinjury ) Personnel Injury               *)
(*  This procedure determines the nature of the personnel injury and      *)
(*  calls the appropriate procedure to provide guidance for emergency     *)
(*  first aid and casualty transportation precautions .                   *)
(******************************************************************************)

procedure perinjury(needrec : recordptr );

var
   anschar   : char ;
   ans,
   tranprec  : integer  ;    (* code for transportation precautions *)
   medemcpt  : boolean  ;    (* used to determine when medical emer over *)

(*#########################################################################*)
(*                   VARIOUS INJURY PROCEDURES                              *)
(*  Procedures headinj, spinalinj, extbleed, intbleed, brokearm,brokeleg, *)
(* severlimb, burns, inhalefumes, electshock, drowning, and unknownill    *)
(* are used to provide guidance in emergency first aid to the emergency   *)
(* response team. One of these procedures will be called by procedure     *)
(* perinjury in the event of a personnel injury.                          *)
(*#########################################################################*)

procedure headinj(var tranprec: integer) ;

begin
     tranprec := 2 ;
end ;  (*  headinj *)

procedure spinalinj(var tranprec: integer );

begin
     tranprec := 2 ;
end ;  (* spinalinj *)

procedure extbleed( var tranprec :integer );

begin
     tranprec := 5 ;

end ; (* extbleed *)

procedure intbleed(var tranprec : integer );

begin
     tranprec := 1 ;

end ; (* intbleed *)

procedure brokeleg(var tranprec : integer);

begin
     tranprec := 4 ;

end ; (* brokeleg *)

procedure brokearm(var tranprec : integer );

begin
     tranprec := 4 ;
end ; (*  brokearm *)

procedure severlimb(var tranprec: integer );
begin
     tranprec := 5 ;
end ;  (* severlimb  *)
```

```
procedure burns( var tranprec :integer );

begin
     writeln(' Try to limit contact and keep the burned area as clean as');
     writeln(' possible. Have the Hospital corpman apply temporary dressings.');
     writeln(' Use wet towels in the stretcher if needed. ');
     tranprec := 1 ;
end ; (* burns  *)

procedure inhalefumes(var tranprec : integer );

begin
     tranprec := 3 ;

end ; (* inhalefumes  *)

procedure electshock( var tranprec :integer );

begin
     tranprec := 3 ;
end ; (* electshock  *)

procedure drowning(var tranprec : integer );

begin
     writeln(' If the victum is unconsious and not breathing have personnel');
     writeln(' on the scence begin mouth-to-mouth breathing immediately.');
     writeln(' Call sickbay and have them deliver oxygen to the victum.  ');
     writeln(' If there is no heartbeat , give external cardiac massage.');
     tranprec := 3 ;
end ;  (*  drowning  *)

procedure unknownill(var tranprec : integer);

begin
     tranprec := 1 ;
end ; (* unknownill *)

(***********************************************************************)
(*              Procedure Transport Precautions (tranport)        *)
(*  This procedure provides guidance for the safe handling of injured  *)
(*  personnel that are to be transported.                        *)
(***********************************************************************)

procedure tranport(tranprec : integer) ;

begin
     writeln;
     case tranprec of
       1 : writeln;

       2 : begin
                writeln(' Immobilize the patient in the stokes stretcher.');
                writeln(' Be careful not to jar or shake the patient. ');
           end;

       3 : writeln(' Maintain CPR during transport if required .');

       4 : begin
                writeln(' If the patient is to be transported on a stretcher, ');
                writeln(' ensure that the temporary cast is properly supported.');
           end;

       5 : writeln(' Maintain pressure on the wound during transportation.');
     end ;  (* case *)

end  ;  (* tranport *)
```

```
(****************************************************************************)
(*                     Procedure Best Route                         *)
(*  This procedure provides the best route to transport injured         *)
(*  personnel to sickbay. This procedure could be developed futher.     *)
(****************************************************************************)

procedure bestroute(needrec :recordptr );

begin
    writeln(' Transport or escort patient to sickbay ',sickbay,' by the ');
    writeln(' closest safe route. ');
    if (needrec@.usecode = 22) and (needrec@.dklevel < > sickbaydk) then
        writeln(' Use Ammo elevators if possible to move injured personnel.');
    if (needrec@.frame < sickbayfr) and (needrec@.shipside = 'S') then
        writeln(' Cross over to the port passageway on the messdecks.');
end ; (* procedure bestroute *)


                (** Start of main procedure personnel injury **)

begin
    medemcpt := false ;
     repeat ;
        writeln(' What type of medical emergency has occurred ?') ;
        writeln(' Enter the appropriate number below : ');
        writeln;
        writeln('    Head Injury              =  1 ');
        writeln('    Back or neck Injury       =  2 ');
        writeln('    External Bleeding         =  3 ');
        writeln('    Internal Injury          =  4 ');
        writeln('    Broken Leg               =  5 ');
        writeln('    Broken Arm                =  6 ');
        writeln('    Severed Limb              =  7 ');
        writeln('    Serious Burns            =  8 ');
        writeln('    Inhalation of Fumes        =  9 ');
        writeln('    Electrical Shock         = 10 ');
        writeln('    Drowning                 = 11 ');
        writeln('    Unknown Illness            = 12 ');
        writeln;
        readln(ans) ;

        case ans of
            1 :  headinj(tranprec) ;
            2 :  spinalinj(tranprec) ;
            3 :  extbleed(tranprec) ;
            4 :  intbleed(tranprec) ;
            5 :  brokeleg(tranprec) ;
            6 :  brokearm(tranprec) ;
            7 :  severlimb(tranprec) ;
            8 :  burns(tranprec) ;
            9 :  inhalefumes(tranprec) ;
           10 :  electshock(tranprec) ;
           11 :  drowning(tranprec) ;
           12 :  unknownill(tranprec) ;
        end;  (* case *)

        tranport(tranprec) ;
        bestroute(needrec) ;
        writeln(' Are there any more medical emergencies ? Y/N ');
        readln(anschar);
        if (anschar = 'N') or (anschar = 'n') then
                medemcpt := true ;
      until medemcpt ;

end ; (* procedure personnel injury *)
```

```
(****************************************************************************)
(*                    Procedure Boundaries                              *)
(* This procedure provides fire, flooding, or fume boundaries to contain *)
(* the hazard to one zone or area of the ship. The procedure also lists  *)
(* the berthing compartments in the area of the ship to be cleared of    *)
(* personnel. The procedure contains procedure berthcomp to determine if *)
(* berthing compartments are in the zone.                                *)
(****************************************************************************)

procedure boundaries(needrec : recordptr ; haztype : emertypes) ;

var
   i,                          (* used as a counter  *)
   temp1, temp2 : integer ;    (* used to manipulate integer data *)
   bertharea : boolean  ;      (* set if berthing comp is in the zone *)

(**             Procedure Berthcomp                    **)

procedure berthcomp(temp1 :integer ; var bertharea :boolean );

begin
   while nextrec < > nil do
      begin
         if (nextrec@.zone = temp1) and (nextrec@.usecode = 2 ) then
            begin
               if not bertharea then
                  writeln(' The following Berthing Compartments should be cleared :');
               writeln(nextrec@.dklevel,' - ',nextrec@.frame,' - ',
                                        nextrec@.loctocl );
               bertharea := true ;
               nextrec := nextrec@.linkit
            end  (* if *)
         else
            nextrec := nextrec@.linkit;
      end ; (* while *)
end ; (* berthcomp *)

                                  (*  start of boundaries  *)
begin
   bertharea := false ;
   temp1 := needrec@.zone ;
   if  not needrec@.compabmndk then
      begin
         nextrec := lwrdk(.needrec@.dklevel.).linkit ;
         berthcomp(temp1, bertharea) ;
         if needrec@.dklevel  <  numdecks then
            begin
               temp2 := needrec@.dklevel  +  1 ;
               nextrec := lwrdk(.temp2.).linkit ;
               temp1 := needrec@.zone  +  10 ;
               berthcomp(temp1, bertharea) ;
            end ; (* if *)
         if needrec@.dklevel  >  1 then
            begin
               temp2 := needrec@.dklevel - 1 ;
               nextrec := lwrdk(.temp2.).linkit ;
               temp1 := needrec@.zone - 10 ;
               berthcomp(temp1, bertharea) ;
            end; (* if *)
      end   (* if not above main deck *)

   else
      begin
         nextrec := uplev(.needrec@.dklevel.).linkit ;
         berthcomp(temp1, bertharea) ;
         if needrec@.dklevel  <  numlevel then
            begin
               temp2 := needrec@.dklevel  +  1 ;
               nextrec := uplev(.temp2.).linkit ;
```

```
            temp1 := needrec@.zone - 10 ;
            berthcomp(temp1, bertharea) ;
         end ; (* if *)
      if needrec@.dklevel > 1 then
         begin
            temp2 := needrec@.dklevel - 1 ;
            nextrec := uplev(.temp2.).linkit ;
            temp1 := needrec@.zone + 10 ;
            berthcomp(temp1, bertharea) ;
         end; (* if *)
   end;  (* else above main deck *)

writeln;
writeln('Set the following  boundaries to contain the hazard :');
writeln;
for i := 0 to numfittings do
   begin
      with closure(.i.) do
         begin
            if zoneloc = needrec@.zone then
               writeln( 'Fitting No. ',fitting,' located ',fitlocat ) ;
         end ;  (* with *)
   end;  (* for *)
continue;

if haztype <> fumes then
  begin
     for i := 0 to numelpanel do
      begin
        with panel(.i.) do
         begin
            if panlcode = needrec@.panelcode then
               begin
                  writeln;
                  writeln('To secure electrical power to compartment ',
                        needrec@.dklevel,' -',needrec@.frame,'-',needrec@.loctocl );
                  writeln('Secure power at panel No. ',panelnum,' located in ',
                                    panlocat ) ;
               end ; (* if *)
         end ;  (* with *)
      end;  (* for *)
  end;  (* if *)

if haztype <> flooding then
  begin
    for i := 0 to numventcon do
      begin
         with vent(.i.) do
           begin
            if vntcode = needrec@.ventcode then
               begin
                  writeln;
                  writeln('To secure ventilation to compartment ',
                        needrec@.dklevel,' -',needrec@.frame,'-',needrec@.loctocl );
                  writeln('Use controller No. ',contnum,' located in ',
                                    contlocat ) ;
               end ; (* if *)
           end ;  (* with *)
      end; (* for *)
  end;   (* if *)
end ;  (* procedure boundaries *)
```

```
(************************************************************************)
(*            Procedure (idensource)  Identify Source                  *)
(*  This procedure provides guidance on methods to determine the source  *)
(*  of the identified hazard.                                           *)
(************************************************************************)

procedure idensource(needrec: recordptr ;haztype:emertypes ;
                                    var hazsource :integer ) ;

var
   ans      : char  ;
   valentry  : boolean ;  (* used to determine if a valid entry was made *)

begin
     valentry := false ;
     writeln;
     if haztype = fire then
          begin
            repeat
               writeln('What type of fire is this ? ');
               writeln('Enter the appropriate number below: ');
               writeln;
               writeln('Class A (paper, wood, materials  etc.)      = 1 ');
               writeln('Class B (hydrocarbon or flammable liquids)  = 2 ');
               writeln('Class C (electrical fire )                 = 3 ');
               writeln('Class D ( Chemical fire )                  = 4 ');
               writeln(' Type of fire unknown                      = 0 ');
               writeln;
               readln(hazsource);
               if (hazsource > 4) or (hazsource < 0 ) then
                    writeln(' Invalid input, please try again. ')
               else
                    valentry := true ;
               if hazsource = 0 then
                    begin
                       writeln('Send out investigators to determine type of fire.');
                       valentry := false ;
                    end ; (* if *)
             until valentry  ;

             writeln(' Is the source of the fire known? Y/N ' );
             readln(ans);
             if (ans = 'N') or (ans = 'n') then
                  begin
                       case hazsource of
                         1 : begin
                                 if needrec@.usecode = 1 then
                                      writeln('Admin space, probable paper fire. ');
                                 if (needrec@.usecode < 10) and (needrec@.usecode > 4) then
                                      writeln('Potential of a class B fire ');
                                 if needrec@.usecode = 3 then
                                      writeln(' Berthing compartment.',
                                        ' Possible bedding fire. Hazardous fumes possible. ');
                             end; (* 1 *)

                         2 : begin
                                 if (needrec@.pipingcode < 13 ) and
                                              (needrec@.pipingcode > 7) then
                                        writeln(' Flammable liquids in piping. ') ;
                                 if needrec@.usecode = 13 then
                                        writeln(' Flammable storeroom ');
                             end ; (* 2 *)

                         3 : writeln(' Check electrical equipment in the space.');

                         4 : begin
                                 if needrec@.usecode = 21 then
                                   writeln('Contact ',needrec@.dklevel,' -',needrec@.frame,
                                             ' -', needrec@.loctocl, ' custodian for detail ',
```

```
                                        'inventory of materials. ');
                              end ; (* 4 *)
                        end;  (* case hazsource *)
                  end;  (* if ans = N  *)
            end ; (*  if  fire  *)

    if haztype =  flooding then
        begin
            repeat
               writeln(' What is flooding the space ? ');
               writeln(' Enter the appropriate number. ');
               writeln;
               writeln('  Salt water       = 1 ' );
               writeln('  Fresh water      = 2 ' );
               writeln('  Chilled water    = 3 ' );
               writeln('  Fuel oil         = 4 ' );
               writeln('  Unknown          = 0 ' );
               writeln;
               readln(hazsource);
               if (hazsource > 4) or (hazsource < 0 ) then
                          writeln(' Invalid input, please try again. ')
               else
                    valentry := true ;
               if hazsource = 0 then
                    begin
                        writeln('Send out investigators to determine type of',
                                                    ' flooding.');
                        valentry := false ;
                    end ; (* if *)
            until valentry  ;

            writeln(' Is the source of the flooding known? Y/N ');
            readln(ans);
            if (ans = 'N') or (ans = 'n') then
                 begin
                     if (hazsource = 1 ) and (needrec@.pipingcode = 3) then
                           writeln(' Check firemain in space ');
                     if (needrec@.dklevel > 3) and ( not needrec@.compabmndk )
                         then
                           writeln(' Check for a leak in the hull or any nearby',
                                                   ' floodable voids. ');
                  end; (* if *)
        end  ;  (* flooding *)

    if haztype = fumes then
        begin
            fumetest(needrec, fumes, hazsource);
            safeprec(fumes,hazsource);
            writeln;
            if (hazsource = 2) and (needrec@.pipingcode = 6) then
                  writeln(' Possible break in plumbing vent. ') ;
            repeat
               writeln(' Have additional fumes been discovered ? Y/N');
               readln(ans);
               if (ans = 'Y') or (ans = 'y') then
                  begin
                     fumetest(needrec, personinj, hazsource);
                     safeprec(fumes, hazsource);
                  end;
            until (ans = 'N') or (ans = 'n')
        end; (* fumes *)

    writeln(' Is the compartment number for the location of the emergency');
    writeln('        correct as originally reported ? Y/N ' );
    readln(ans);
    if (ans = 'N') or (ans = 'n') then
          getcompnum(needrec) ;
  end ;  (* procedure idensource *)
```

```
(***********************************************************************)
(*              Procedure (sysguide) System Guidance              *)
(* This procedure provides guidance on the proper operation of the    *)
(* installed damage control system that is inputed by the parameter   *)
(* systmnum                                                        *)
(***********************************************************************)

procedure sysguide(systmnum : integer) ;

begin
    writeln;
    case systmnum of
     1 : begin
            writeln(' To operate the installed eductor system, open the (OVBD)');
            writeln(' overboard valve first, then the activating water (ACTW),');
            writeln(' and finally open the bilge or main drain (SUCT) valve.');
            writeln(' To secure the eductor use the reverse order of lineup.');
         end;  (*  1 *)

     2 : begin
            writeln(' To operate the installed exhaust fan, open any closed ');
            writeln(' exhaust vent dampers and then activate the exhaust fan ,');
            writeln(' at the given controller. In the event of a fire, secure');
            writeln(' the fan at the given controller and then close the ');
            writeln(' exhaust vent dampers. ');
         end;  (*  2 *)

     3 : begin
            writeln(' To operate the active sprinkler system, open the ');
            writeln(' given control valve . ');
            writeln(' To secure the sprinker close the control valve or use ',
                          'the COV. ');
         end;  (*  3 *)

     4 : begin
            writeln(' To use the dry standpipe , connect a firehose to the ');
            writeln(' given fitting. For plain saltwater use a fire station.');
            writeln(' For AFFF use a portable FP-180 proportioner.');
         end;  (*  4 *)

     5 : begin
            writeln(' Before operating the installed CO2 flooding system,',
                    ' first ensure that');
            writeln(' all personnel are out of the compartment to be flooded.,');
            writeln(' All supply and exhaust ventilation should be secured. ');
            writeln(' All opening to the compartment should be closed. ');
            writeln(' To operate the CO2 flooding system, pull the given control'
            writeln(' handle firmly enough to break the seals on the CO2 cylinders
         end;  (*  5 *)

     6 : begin
            writeln('Before operating the installed CO2 hose reel system, ');
            writeln('pull the given control handle or lever firmly enough to ');
            writeln('break the seals on the CO2 cylinders. Use the horn in a');
            writeln('sweeping motion directing the CO2 at the base of the fire.');
            writeln('Use the bail on the horn to control the flow of CO2. ');
            writeln('For futher information look at NAVSEA tech manual',
                          ' Chapter 9930  Firefighting -Ship .');
         end;  (*  6 *)

     7 : begin
            writeln(' Before operating the installed HALON system , all supply');
            writeln(' and exhaust vent dampers should be closed if possible, ');
            writeln(' and all fans serving the space secured. All personnel ');
            writeln(' should be out of the space or in the process of leaving');
            writeln(' the space. To activate the HALON system use the given');
            writeln(' local or remote pneumatic actuators. Breathing apparatus');
            writeln(' should be used by personnel entering a space flooded ',
                          'with HALON.');
```

```
          end;  (*  7  *)

      8 : begin
                writeln(' To operate the Twin Agent Fire Extinguishing System , ');
                writeln(' remove the safety clip and pull the give nitrogen cylinder')
                writeln(' valve lever. This activates both the AFFF and PKP systems');
                writeln(' through an interlock valve. Once activated, either agent');
                writeln(' discharge may be shut off at any time by opening or closing'
                writeln(' the pistol grip shut off nozzles.  ');
                writeln(' To secure the TAFES system follow the procedures given in ')
                writeln(' NAVSEA Tech Manual  Chapter 9930, Firefighting-Ship. ');
          end;  (*  8 *)

      9 : begin
                writeln(' To operate the AFFF sprinkler system, operate ');
                writeln(' given fittings. Ensure that the Fog Foam station is ');
                writeln(' manned by personnel who will keep the tank filled.');
          end;  (*  9 *)

     10 : begin
                writeln(' To operate the AFFF Hose Reel System , open');
                writeln(' the given control valve . ');
                writeln(' To secure the Hose Reel follow the procedures given in ');
                writeln(' NAVSEA Tech Manual  Chapter 9930, Firefighting-Ship. ');
          end;  (*  10 *)
    end ;  (* case *)
    writeln;
    continue;
end ;  (* sysguide *)

(*###########################################################################*)
(*                GET SYSTEM INFORMATION PROCEDURES                           *)
(*   The following ten procedures are called by procedures elimhazard and   *)
(* recover to obtain information on the identification number and location  *)
(* of the controls that operate the applicable installed damage control     *)
(* systems and equipment. These procedures call procedure sysguide.         *)
(***************************************************************************)

procedure geteduct(needrec : recordptr) ;

var
    i  : integer ;   (* used as an index *)

begin
    writeln(' The compartment can be dewatered using the main drainage system.');
    for i :=  1 to numeduct do
        begin
          with eductor(.i.) do
            begin
              if (needrec@.dklevel  =  dkorlev ) and (needrec@.frame  =  fr) and
                     (needrec@.loctocl  =  thirdpt) then
                 writeln(' Valve number ',valvenum,' located ',location);
            end; (* with *)
        end ; (*  for  *)
    sysguide(1) ;
end;  (* geteduct *)

(********************************************)
procedure getexhaust(needrec : recordptr) ;

var
    i  : integer ;   (* used as an index *)

begin
    writeln(' The compartment has an installed exhaust system.');
    for i :=  1 to numexhfn do
        begin
          with exhfan(.i.) do
            begin
```

```
                if (needrec@.dklevel = dkorlev ) and (needrec@.frame = fr) and
                      (needrec@.loctocl = thirdpt) then
                   writeln(' Controller number ',valvenum,' located ',location);
             end; (* with *)
         end ; (* for *)
      sysguide(2) ;
end;    (* getexhaust *)

(**********************************************)
procedure getactspr(needrec : recordptr) ;

var
      i : integer ;   (* used as an index *)

begin
      writeln(' The compartment contains an active sprinkler system.');
      for i := 1 to numactsp do
         begin
            with actspr(.i.) do
             begin
                if (needrec@.dklevel = dkorlev ) and (needrec@.frame = fr) and
                      (needrec@.loctocl = thirdpt) then
                   writeln(' Valve number ',valvenum,' located ',location);
             end; (* with *)
         end ; (* for *)
      sysguide(3) ;
end;  (* getactspr *)

(*******************************************)
procedure getspspr(needrec: recordptr) ;

var
      i : integer ;   (* used as an index *)

begin
      writeln(' The compartment contains a dry standpipe sprinkler system.');
      for i := 1 to numdysp do
         begin
            with drystd(.i.) do
             begin
                if (needrec@.dklevel = dkorlev ) and (needrec@.frame = fr) and
                      (needrec@.loctocl = thirdpt) then
                   writeln(' Valve number ',valvenum,' located ',location);
             end; (* with *)
         end ; (* for *)
      sysguide(4) ;
end;    (* get stand pipe sprinkler  *)

(*******************************************)
procedure getco2fld(needrec : recordptr) ;

var
      i : integer ;   (* used as an index *)

begin
      writeln(' The compartment contains a CO2 flooding system.');
      for i := 1 to numcofld do
         begin
            with co2fld(.i.) do
             begin
                if (needrec@.dklevel = dkorlev ) and (needrec@.frame = fr) and
                      (needrec@.loctocl = thirdpt) then
                   writeln(' Valve number ',valvenum,' located ',location);
             end; (* with *)
         end ; (* for *)
      sysguide(5) ;
end;  (* getco2fld *)

(*********************************************)
```

```
procedure getco2hr(needrec : recordptr) ;

var
     i  : integer ;   (* used as an index *)

begin
     writeln(' The compartment contains a CO2 Hose Reel system.');
     for i :=  1 to numcohr do
         begin
            with co2hr(.i.) do
                begin
                   if (needrec@.dklevel  =  dkorlev ) and (needrec@.frame  =  fr) and
                          (needrec@.loctocl  =  thirdpt) then
                       writeln(' Valve number ',valvenum,' located ',location);
                end; (* with *)
         end ; (*  for  *)
     sysguide(6) ;
end;   (*  getco2hr *)

(**********************************************)
procedure getafffsp(needrec : recordptr) ;

var
     i  : integer ;   (* used as an index *)

begin
     writeln(' The compartment contains a AFFF sprinkler system.');
     for i :=  1 to numafffsp do
         begin
            with afffspr(.i.) do
                begin
                   if (needrec@.dklevel  =  dkorlev ) and (needrec@.frame  =  fr) and
                          (needrec@.loctocl  =  thirdpt) then
                       writeln(' Valve number ',valvenum,' located ',location);
                end; (* with *)
         end ; (*  for  *)
     sysguide(9) ;
end;  (* getafffsp *)

(**********************************************)
procedure gethalon(needrec : recordptr) ;

var
     i  : integer ;   (* used as an index *)

begin
     writeln(' The compartment°s HALON actuators are as follows :');
     for i :=  1 to numhalon do
         begin
            with halon(.i.) do
                begin
                   if (needrec@.dklevel  =  dkorlev ) and (needrec@.frame  =  fr) and
                          (needrec@.loctocl  =  thirdpt) then
                       writeln(' Actuator  ',valvenum,' located ',location);
                end; (* with *)
         end ; (*  for  *)
     sysguide(7) ;
     if needrec@.usecode  =  6 then
          begin
             writeln(' Activate AFFF Bilge sprinkler system. ');
             getafffsp(needrec);
          end;
end;  (* gethalon *)

(**********************************************)
procedure gettafes(needrec: recordptr) ;

var
     i  : integer ;   (* used as an index *)
```

```
begin
    writeln(' The compartment contains a Twin Agent Extinguishing System.');
    for i := 1 to numtau do
        begin
          with tausys(.i.) do
              begin
                if (needrec@.dklevel = dkorlev ) and (needrec@.frame = fr) and
                        (needrec@.loctocl = thirdpt) then
                    writeln(' Valve number ',valvenum,' located ',location);
              end; (* with *)
        end ; (* for *)
    sysguide(8) ;
end;    (* gettafes *)


(*********************************************)
procedure getafffhr(needrec : recordptr) ;

var
    i : integer ;  (* used as an index *)

begin
    writeln(' The compartment contains a AFFF Hose Reel system.');
    for i := 1 to numafffhr do
        begin
          with afffhr(.i.) do
              begin
                if (needrec@.dklevel = dkorlev ) and (needrec@.frame = fr) and
                        (needrec@.loctocl = thirdpt) then
                    writeln(' Valve number ',valvenum,' located ',location);
              end; (* with *)
        end ; (* for *)
    sysguide(10) ;
end;    (* getafffhr *)

(**********************************************************************)
(*            Procedure (elimhazard) Eliminate Hazard            *)
(*  This procedure provides guidance on methods to eliminate the    *)
(*  identified hazard and the use of applicable installed damage    *)
(*  control equipment.                                               *)
(*  This procedure calls procedure safeprec and one of the ten GET  *)
(*  SYSTEM procedures as appropriate.                                *)
(**********************************************************************)

procedure elimhazard(needrec : recordptr; haztype : emertypes;
                                                   hazsource : integer );

var
   ans       : char    ; (* used to read in answers from user *)
   systmnum  : integer ; (* used to identify installed systems *)


                                        (*  start of procedure elimhazd *)
begin
    if haztype = fire then
        begin
          safeprec(fire, hazsource);
          if (needrec@.usecode < 10) and (needrec@.usecode > 5) then
              begin
                writeln(' Secure the exhaust ventilation to the space. ');
                getexhaust(needrec);
              end;  (*  if  *)

          if (needrec@.instalsystems < 3) and (hazsource < 3) then
                begin
                  writeln('No appropriate installed systems, use nearest',
                                                   ' fire station.')  ;
                  if hazsource = 2 then
```

```
            begin
              writeln(' Use portable FP-180 Foam Proportioner or use ');
              writeln(' portable PKP extinguishers.');
            end;
      end;

  if hazsource = 3 then
      begin
        if (needrec@.instalsystems < 5) or ((needrec@.instalsystems > 7)
                  and (needrec@.instalsystems <> 11)) then
            writeln('No appropriate installed systems, use nearest',
                          portable CO2 extinguishers.')
      end;

  if (needrec@.instalsystems < 5) and (hazsource = 4) then
        writeln('No appropriate installed systems, use nearest',
                         ' appropriate portable equipment.');

  case needrec@.instalsystems of

    3 : if hazsource < 3 then
              getactspr(needrec);

    4 : if hazsource < 3 then
              getspspr(needrec);

    5 : getco2fld(needrec);

    6 : begin
            if hazsource = 1 then
                writeln('Use fire station to combat class "A" fire',
                                           ' if possible .');
            if hazsource = 2 then
                begin
                  writeln('Use fire station or Portable PKP ',
                                  'extinguishers to');
                  writeln(' combat class "B" fire if possible.');
                end;
            if hazsource = 4 then
                  writeln('Use CO2 if appropriate to ',
                                 'extinguish the class "D" fire.');
            getco2hr(needrec);
          end;  (* 6 *)

    7 : begin
            writeln(' Use the HALON system if the fire is out',
                                       ' of control or great');
            writeln(' hazards to personnel are involved in',
                                  ' trying to combat the fire .');
            gethalon(needrec);
          end; (* 7*)

    8 : if hazsource <> 3 then
            begin
              if hazsource = 4 then
                  writeln(' Use TAFES to extinguish class "D" fire',
                                  ' if appropriate.');
                  gettafes(needrec);
            end; (* 8 *)

    9 : if hazsource < 3 then
              getafffsp(needrec);

  10 : if hazsource < 3 then
              getafffhr(needrec);

  11 : begin
            writeln('The compartment contains various ',
                                  'firefighting systems.');
```

```
                writeln(' Use the most appropriate system.');
                if hazsource = 4 then
                     writeln(' Use TAFES to extinguish class "D" fire',
                                    ' if appropriate.');
                if hazsource < > 3 then
                     gettafes(needrec);
                if hazsource < 3 then
                   begin
                      getafffsp(needrec);
                      getafffhr(needrec);
                   end;  (* if *)
                gethalon(needrec);
             end;  (* 11 *)

      12 : begin
                if hazsource = 4 then
                     writeln(' Use AFFF to extinguish class "D" fire',
                                    ' if appropriate.');
                getafffhr(needrec);
                if hazsource < > 2 then
                     getco2hr(needrec) ;
             end; (* 12 *)

     end;  (* case instalsystem of *)
   end ;  (* if fire *)

if haztype = flooding then
   begin
     safeprec(flooding , hazsource);
     case hazsource of
      1 : begin  (* saltwater  *)
               if (needrec@.pipingcode = 1) or (needrec@.pipingcode = 3) then
                     writeln(' If the source of flooding is the firemain,');
                     writeln(' isolate the section of firemain.');
            end;  (* saltwater *)

      2 : begin  (* freshwater  *)
               if (needrec@.pipingcode = 2) or (needrec@.pipingcode = 3) then
                     writeln(' If the source of flooding is the fresh water ');
                     writeln(' piping isolate the section of piping.');
            end;  (* freshwater *)

      3 : begin  (* chilled water  *)
               if (needrec@.pipingcode = 4) or (needrec@.pipingcode = 12) then
                     writeln(' If the source of flooding is chilled water ,',
                               ' isolate the section of chilled water piping.');
            end;  (* chilled water *)

      4 : begin  (* fuel oil  *)
               if (needrec@.pipingcode > 7) and (needrec@.pipingcode < 13) then
                     writeln(' If the source of flooding is the fuel system,',
                               ' isolate the section of fuel piping.');
            end;  (* fuel oil *)
     end; (* case hazsource of *)

     if (needrec@.instalsystems < > 1) or (needrec@.instalsystems < > 10)
             or (needrec@.instalsystems < > 11)
                  or (needrec@.instalsystems < > 12)  then
        begin
          writeln('No appropriate installed dewatering systems, use portable');
          writeln('dewatering equipment.' );
          if hazsource = 4 then
             writeln(' Do not use a P-250 pump to pump oil or fuel.');
        end
     else
          begin
             case needrec@.instalsystems of
                1 : geteduct(needrec);
                8 : geteduct(needrec);
```

```
                9 : geteduct(needrec);
               10 : geteduct(needrec);

               11 : begin
                        writeln('The compartment contains various ',
                                        'dewatering systems.');
                        writeln(' Use the most appropriate system.');
                        geteduct(needrec);
                        writeln(' The main circulation pump for the main',
                                    ' condensor can be used to ');
                        writeln('dewater the bilges of a Main Machinery Room.');
                    end;  (* 11 *)

               12 : geteduct(needrec);
              end;  (* case instalsystem of *)
          end; (* else installed systems *)

    end ;  (* flooding *)

if haztype = fumes then
    begin
      case hazsource of
        1 : begin  (* Explosive gases  *)
               if (needrec@.pipingcode = 8) or (needrec@.pipingcode = 12) then
                 begin
                   writeln(' A possible source of the fumes is the fuel piping.');
                   writeln(' Isolate all fuel and fuel tank vent lines.');
                 end;
               writeln(' Use a pneumatic blower to remove explosive gases.');
            end;  (* explosive gases *)

        2 : begin  (* Hydrogen Sulfide  *)
               if (needrec@.pipingcode = 5) or (needrec@.pipingcode = 6) then
                   begin
                     writeln(' Close all COV for all soil lines and plumbing');
                     writeln(' vents leading to the space.');
                   end;
            end;  (* Hydrogen Sulfide *)

        3 : begin  (* Benzine  *)
               if (needrec@.instalsystems = 2) or (needrec@.instalsystems = 12)
                      or (needrec@.instalsystems = 11)  then
                   getexhaust(needrec);
            end;  (* Benzine *)

        4 : begin  (* Chlorine *)
               if (needrec@.instalsystems = 2) or (needrec@.instalsystems = 12)
                      or (needrec@.instalsystems = 11)  then
                   getexhaust(needrec);
            end;  (* Chlorine *)

        5 : begin  (* Carbon monoxide  *)
               if (needrec@.dklevel > 3) and ( not needrec@.compabmndk ) then
                   writeln(' Check for nearby open or damaged voids.');
               if (needrec@.instalsystems = 2) or (needrec@.instalsystems = 12)
                      or (needrec@.instalsystems = 11)  then
                   getexhaust(needrec);
            end;  (* Carbon Monoxide  *)

        6 : begin  (* Carbon Dioxide  *)
               if (needrec@.instalsystems = 2) or (needrec@.instalsystems = 12)
                      or (needrec@.instalsystems = 11)  then
                   getexhaust(needrec);
            end;  (* Carbon Dioxide  *)

      end; (* case hazsource of *)
    end ;  (* if fumes *)
end;  (* elimhazard *)
```

```
(***************************************************************************)
(*                     Procedure Recover                     *)
(* This procedure provides guidance on the proper methods to recover   *)
(* from the identified emergency. It provides recommended tests to     *)
(* conduct to determine if the compartment is safe to enter without    *)
(* breathing devices. The procedure contains procedures ventilate and  *)
(* dewater to provide the control numbers and locations of appropriate *)
(* ventilation and dewatering systems.                                 *)
(***************************************************************************)

procedure recover(needrec : recordptr ; haztype : emertypes);

var
     ans  : char    ;     (* used to read in answers from the user *)
     fumetype : integer ; (* used to determine the type of fumes *)
     hazdelim : boolean ; (* used to determine if hazard eliminated *)

(*****************************************************)
procedure ventilate(needrec : recordptr; fumetyp : integer);

begin
     writeln;
     if fumetyp = 1 then
        writeln(' Recommend use of portable pneumatic blowers to remove ',
                         'explosive vapors.');
     if (needrec@.instalsystems < > 2) and (needrec@.instalsystems < 5 ) then
        begin
           writeln(' No installed exhaust ventilation systems.');
           writeln(' Use a portable pneumatic blower or a safety checked ',
                         'Red Devil blower.');
        end   (* if *)
     else
          getexhaust(needrec);
     writeln;
end;  (* ventilate *)

(*******************************)
procedure dewater(needrec : recordptr );

begin
     writeln;
     if needrec@.usecode = 6 then
          writeln('The main circ pump can be used to dewater a Main ',
                              'Machinery space.');
     if (needrec@.instalsystems = 1) or (needrec@.instalsystems > 7) then
           geteduct(needrec)
     else
        begin
           writeln(' No installed dewatering equipment in the space.');
           writeln(' Use portable equipment utilizing the nearest overboard',
                              ' discharge');
           writeln(' Use an eductor if the water contains fuel or debris.');
           writeln(' An electrical submersible pump can be used if the water');
           writeln(' is fairly clear of oil and debris. A P-250 pump can be');
           writeln(' used if adequate ventilation to remove exhaust fumes is',
                              ' available.' );
        end ; (* else *)
end ; (*  dewater  *)

                              (* start of procedure recover *)
begin
   if haztype = fire then
       begin
         hazdelim := false ;
         repeat
           writeln;
           writeln(' Is the fire out ? Y/N' );
           writeln;
```

```
        readln(ans);
        if (ans= 'N') or ( ans = 'n') then
            writeln(' Continue to combat the fire until it is extinguished.')
        else
            begin
              hazdelim := true ;
              fumetest( needrec, fire, fumetype);
              if fumetype = 0 then
                begin
                  writeln(' Compartment ',needrec@.dklevel,'-',needrec@.frame,
                          '-', needrec@.loctocl,' is safe to enter .');
                  writeln(' Set the reflash watch and overhaul the fire.');
                  writeln;
                end
              else
                begin
                  safeprec(fumes, fumetype);
                  writeln('Set the reflash watch in OBAs and overhaul the fire.');
                  ventilate(needrec , fumetype);
                end; (* else < > 0 *)
              writeln(' Is the fire overhauled ? Y/N ');
              writeln;
              readln(ans) ;
              if (ans= 'N') or ( ans = 'n') then
                begin
                    writeln(' Continue to overhaul the fire . Conduct second ');
                    writeln(' test when the overhaul is completed.')
                end;
              fumetest( needrec, fire, fumetype);
              if fumetype = 0 then
                begin
                  writeln(' Compartment ',needrec@.dklevel,'-',needrec@.frame,
                          '-', needrec@.loctocl,' is safe to enter .');
                  writeln(' No ventilation is needed except for Smoke removal');
                  writeln(' Does the compartment need to be desmoked ? Y/N');
                  writeln;
                  readln(ans) ;
                  if (ans= 'Y') or ( ans = 'y') then
                          ventilate(needrec , fumetype);
                  writeln;
                end
              else
                begin
                  safeprec(fumes, fumetype);
                  ventilate(needrec, fumetype);
                end; (* else < > 0 *)
              if (needrec@.instalsystems < > 1) and (needrec@.instalsystems < 9)
               then begin
                 writeln('Does Compartment ',needrec@.dklevel,'-',needrec@.frame,
                          '- ', needrec@.loctocl,' have a significant amount of');
                 writeln(' water, over two inches, that needs to be removed .');
                 writeln;
                 readln(ans) ;
                 if (ans= 'N') or ( ans = 'n') then
                   begin
                     writeln(' Recommend the use of mops and pails to clean up.');
                     writeln(' Use the closest deck drains to remove the water.')
                   end
                 else
                   dewater(needrec);
               end   (* if then begin *)
             else
               dewater(needrec);
           end; (* else fire out *)
      until hazdelim ;
    end; (* haztype = fire *)

  if haztype = flooding then
    begin
```

```
          dewater(needrec) ;
          writeln(' Ensure all electrical equipment and wiring is checked out');
          writeln(' before restoring electrical power to the compartment. ');
       end;  (* haztype = flooding *)

    if haztype = fumes then
       begin
          writeln(' Check compartment to ensure fumes are cleared from space.');
          fumetest(needrec, fumes, fumetype);
          if fumetype = 0 then
             begin
                writeln(' Compartment ',needrec@.dklevel,'-',needrec@.frame,
                        '-', needrec@.loctocl,' is safe to enter .');
                writeln(' No ventilation is needed . ');
                writeln;
             end
          else
               begin
                  safeprec(fumes, fumetype);
                  ventilate(needrec, fumetype);
               end; (* else < > 0 *)
       end;  (* haztype = flooding *)
end ; (* procedure recover *)

              (**** Start of main procedure  natknown ****)

begin
     writeln(' What is the nature of the emergency or problem ? ' );
     writeln(' Enter the appropriate number below :  ') ;
     writeln;
     writeln('    Fire                  = 1  ') ;
     writeln('    Flooding              = 2  ') ;
     writeln('    Fumes                 = 3  ') ;
     writeln('    Injury to personnel   = 4  ') ;
     writeln;
     readln(hazcode) ;
     case hazcode of
        1 : haztype := fire ;
        2 : haztype := flooding ;
        3 : haztype := fumes ;
        4 : haztype := personinj ;
     end ; (* case *)

     if haztype = personinj then
          perinjury(needrec)
     else
        begin
           boundaries(needrec ,haztype) ;
           idensource(needrec, haztype, hazsource);
           elimhazard(needrec, haztype, hazsource);
           recover(needrec, haztype );

        end ; (* else *)

end ;  (* procedure natknown  *)
```

(***** Start of Main Program *****)

```
begin
   writeln('************************************************************************');
   writeln('*                         EMERGENCY                              *');
   writeln('*  An Expert system to provide guidance on the operation of  *');
   writeln('*  installed damage control equipment aboard the ship.       *');
   writeln('*  Answer the given questions and make the appropriate       *');
   writeln('*  choices from the presented  menus.                        *');
   writeln('************************************************************************');
   writeln;
   emerover := false  ;
   loadfiles ;
   loaddata ;
   writeln(' Do you want to have the ship°s compartments printed out ? Y/N ');
   writeln;
   readln(answer);
   if (answer = 'Y') or (answer = 'y') then
          printcomp ;

   repeat  ;        (* Until all emergencies have been handled  *)
   getcompnum(needrec) ;
   writeln;
   writeln('Is the nature of the emergency known ? Y/N ');
   writeln;
   readln(answer);
   if (answer = 'N') or (answer = 'n') then
      begin
         unkwnature(needrec) ;
         natknown(needrec);
      end
   else
        natknown(needrec)  ;
   writeln;
   writeln('Are there any more emergencies or casualities ? Y/N' );
   writeln;
   readln(answer);
   if (answer = 'N') or (answer = 'n') then
      emerover := true ;

   until emerover ;

end.  (** Program Emergency  **)
```

LIST OF REFERENCES

1. Naval Sea Systems Command Report S9086-Cn-STM-20, *Naval Ship's Technical Manual, Chapter 079, Volume 2, Damage Control - Practical Damage Control*, December 1979.

2. Bissel, A. M. , Oertel, E. J., and Livingston, D. J., *Shipboard Damage Control*, Naval Institute Press, 1976.

3. Hayes-Roth, F., Waterman, D. A., and Lenat, D. B., *Building Expert Systems*, Addison-Wesley Publishing Company Inc., 1983.

4. Naval Sea Systems Command Report 0901-LP-930-0003, *Naval Ships Technical Manual, Chapter 9930, Firefighting - Ship*, December 1985.

5. Weingart, Stephen G., *Development of a Shipboard Damage Control Fire Team Leader Intelligent Computer Aided Instructional Tutoring System*, Master's Thesis, Naval Postgraduate School, Monterey, California, June 1986.

6. Rowe, N. C., *Introduction to Artificial Intelligence Through Prolog*, Prentice-Hall, 1987.

7. Koffman, Elliot B., *Problem Solving And Structured Programming in Pascal*, Addison-Wesley Publishing Company Inc., April 1982.

8. Borland International, *TURBO PASCAL, Version 3.0 Reference Manual*, 1985.

9. Joy, E. M., Graham, S. L., Haley, C. B., McKusick, M. K., and Kessler, P. B., *BERKELEY PASCAL User's Manual, Version 3.0*, University of California, Berkeley, July 1985.

10. MacLennan, Bruce J, *Principles of Programming Languages: Design, Evaluation, and Implementation*, Holt, Rinehart, and Winston, 1983.

BIBLIOGRAPHY

Naval Sea Systems Command Report 9086-SX-STM-020, *Naval Ship's Technical Manual, Chapter 550, Industrial Gases Generating, Handling, and Stowage*, September 1982.

Naval Sea Systems Command Report S9086-WK-STM-000, *Naval Ship's Technical Manual, Chapter 670, Stowage, Handling, and Disposal of Hazardous General Use Consumables*, February 1976.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5002	2
3.	Department Chairman, Code 52 Department of Computer Science Naval Postgraduate School Monterey, California 93943-5000	1
4.	Computer Technology Programs, Code 37 Naval Postgraduate School Monterey, California 93943-5000	1
5.	Associate Professor N.C. Rowe Code 52Rp Department of Computer Science Naval Postgraduate School Monterey, California 93943-5000	1
6.	Professor Gordon H. Bradley Code 52Bz Department of Computer Science Naval Postgraduate School Monterey, California 93943-5000	1
7.	Commanding Officer Naval Research Laboratory Washington, DC 20375	1
7.	Commanding Officer Naval Damage Control Training Center Philadelphia, PA 19112	1
8.	LCDR Bernard G. Gogel Rural Route 3, Box 197 Huntingburg, Indiana 47542	2